INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Autoclipping

JOSÉ MARIA PAIVA JESUS OLIVEIRA
Outubro de 2020

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Autoclipping: Automatic gathering of news for a specific topic taxonomy

José Oliveira

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Mestrado de Informática

Orientador: Ricardo Almeida
Co-Orientador: Nuno Escudeiro

Porto, 15 de Outubro de 2020

Dedicatória

Ao meus orientadores Professor Doutor Ricardo Almeida (RAL) e Professor Doutor Nuno Escudeiro quero deixar os meus mais sinceros agradecimentos por todo o apoio, orientação e disponibilidade que me deram ao longo destes meses.

Quero agradecer também à minha família e amigos por todo o apoio que me deram para chegar até aqui.

Resumo

A monitorização dos media com o objetivo de compilar notícias sobre determinado assunto, processo denominado de *clipping*, procura cada vez mais recursos à medida que aumenta a quantidade de informação *online*. Usar soluções de aprendizagem automática para auxiliar os editores de boletins temáticos pode ser uma maneira muito eficiente de oferecer suporte ao recorte automático na *web*. Este documento apresenta soluções para a recolha automática de páginas *web* de *seed websites* de interesse para recolher notícias potencialmente interessantes para o boletim da European Association of ERASMUS Coordinators. O processo de recolha retorna dados não estruturados que são pré-processados para que possam ser explorados por técnicas de aprendizagem automática. Em particular, usaremos classificadores de texto para rotular notícias recentes sobre uma taxonomia que representa o tópico de interesse. O *web crawling* que faz a recolha de notícias também recolhe estatísticas sobre a qualidade das notícias extraídas de cada *seed websites* para que o modelo possa adaptar automaticamente a sua frequência de rastreamento para evitar o desperdício de recursos ao extrair dados de sites estáticos. A avaliação preliminar mostra que esse processo pode recolher notícias valiosas com uma redução significativa no tempo e no esforço exigidos do editor do boletim informativo.

Palavras-chave: *Web Crawling*, *Text Mining*, Aprendizagem supervisionada, Classificação.

Abstract

Monitoring the media with the purpose of compiling news about a certain topic, a process named clipping, demands for more and more resources as the amount of online information grows. Using machine learning solutions to assist the editors of thematic newsletters might be a very efficient way to support automatic clipping on the web. This document presents solutions for the automatic harvesting of web pages from seed websites of interest to gather potentially interesting news for the newsletter of the European Association of ERASMUS Coordinators. The harvesting process returns unstructured data that is pre-processed so it can be explored by machine learning techniques. In particular, we will use text classifiers to label fresh news on a taxonomy representing the topic of interest. The web crawler doing the news harvesting is also collecting statistics about the quality of the news extracted from each seed website so the model can automatically adapt its crawling frequency to avoid wasting resources retrieving data from static websites. The preliminary evaluation shows this process might collect valuable news with a significant reduction in the time and effort required from the newsletter editor.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A European Association of ERASMUS Coordinators (EAEC) faz uma exploração dos conteúdos publicados no último mês em *websites* de interesse para a produção do seu boletim mensal. O processo de pesquisa e análise de informação é feito de forma manual pelos seus colaboradores. A EAEC pretende automatizar a recolha de informação e disponibilizá-la em diretorias associadas a um tópico. A diminuição do tempo despendido na recolha de notícias, permitirá uma redução significativa na duração da produção do boletim.

Atualmente, existe um elevado número de artigos de informação e estes abordam variados temas. Uns são mais relevantes para um certo tipo de individuo ou grupo de indivíduos comparativamente a outros. Desde os primórdios da imprensa escrita, constatou-se que nem todas as pessoas folheavam o jornal na sua totalidade, focando-se apenas na secção de notícias do seu interesse (Popp 2014).

Mae Carr é conhecida como a primeira profissional a marcar páginas de jornais relevantes para os seus clientes - governadores, senadores, homens de negócio -, em 1814. No final do século XIX, os jornais apercebendo-se da procura do mercado, começaram a explorar este ramo, vendendo os artigos individualmente ou em volumes. Um novo ramo de negócio dedicado exclusivamente à extração de informação e sua revenda, surgiu (Popp 2014).

Os avanços tecnológicos têm promovido o rápido crescimento da quantidade de informação. A IBM estima que por dia sejam gerados 2.5 triliões de bytes. É ainda estimado que em 2020, 95% dos dados esteja sobre a forma de diferentes formatos de informação não estruturada (textos, vídeos, imagens, áudio, websites, entre outros). A maioria da informação existente é disponibilizada numa formatação não padronizada e proveniente de diversas fontes (Adnan e Akbar 2019). O cenário descrito, torna a tarefa de pesquisa de certos tópicos tediosa pois, para além de se encontrar a informação em diferentes locais, esta pode estar introduzida num enquadramento mais global.

De um modo geral, nas últimas décadas aumentou o nível de atividade de tecnologias para a análise de números, bem como de outros dados estruturados, como por exemplo o país, a idade e as datas. Hoje já se encontram no mercado algumas soluções com resultados convincentes. O *business intelligence* é um dos produtos de maior relevância que chega ao utilizador, revelando informação bastante relevante para o negócio construída a partir dos dados das transações das operações da empresa (Jourdan, Rainer e Marshall 2008).

Fazendo uso das tecnologias de computação, a análise de texto teve início nos finais da década de 90, surgindo o *text data mining* ou *text mining*. Esta área começou por procurar um conjunto de palavras, passando depois a uma fase em que contava o número de vezes que certos termos apareciam, de modo a conseguir categorizar um artigo (Know 2012).

O *text mining* tem auxiliado as organizações na análise do *feedback* dado pelos clientes. Com a expansão das redes sociais, os consumidores têm por norma expressar a sua reação a determinados produtos, campanhas e reviews, sendo este um novo meio em forte expansão, no qual as entidades procuram analisar os sentimentos do público de modo a perceberem a sua reação. Nas caixas de correio eletrónicas o uso de *text mining* é muito usado, tendo sido impulsionado por causa das mensagens de correio eletrónico marcadas como *spam*, permitindo assim que conteúdos impróprios sejam filtrados automaticamente logo após a receção. Por fim, a última tendência tem sido o uso destas tecnologias para resumir artigos, o que facilita o trabalho aos profissionais que procuram e analisam documentos com muito texto (*Text Mining: What is text mining and how it can be useful in Analytics* 2020).

1.1 Problema

A European Association of ERASMUS Coordinators (EAEC)[1] é composta por 140 membros, que de entre outras coisas, colaboram mensalmente para a produção de um boletim temático. A rede produz conteúdo à volta do programa ERASMUS+ e de outros projetos europeus como, por exemplo, a promoção da deslocação dos alunos na União Europeia, organização de seminários e propostas para serem submetidas na Comissão Europeia. Dada a diversidade e abundância de temas, alguns membros, associados e/ou simples subscritores, expressaram vontade em ter uma ferramenta que possibilitasse procurar na *Internet* possíveis notícias de interesse. Tal ambição, deu origem ao projeto proposto: um conjunto de ferramentas automáticas que permitam procurar, recolher, analisar e guardar novas publicações num diretório acessível ao interessado. Os algoritmos desenvolvidos deverão ser eficientes ao analisar o conteúdo. Com um diretório com notícias pré-selecionadas, os associados da EAEC deixariam de perder tempo na procura de informações, sobrando mais tempo para a redação do seu boletim.

Na procura de informação sobre um determinado tema na *Internet*, geralmente são apresentadas milhares de páginas *web*. As fontes de informação e o número de textos publicados, aumentam diariamente. As redes sociais agravaram esta situação, o seu crescimento exponencial expõe muito conteúdo na *web*, sendo que na maioria das vezes contêm informações pouco relevantes. A tarefa de procurar e analisar informação sobre um determinado tema é um processo moroso. O recurso humano encarregue desta tarefa tem ainda de ser conhecedor do tema, pelo que dada a complexidade da função será um profissional com remunerações mais elevadas.

As novas tecnologias trazem uma maior comodidade na resolução das tarefas. No entanto, na procura de informação, a indicação das fontes é na maioria dos algoritmos indicada manualmente. Quando é realizada automaticamente, utiliza mecanismos que podem direcionar o programa informático para páginas com algum ou nenhum relevo, utilizando algumas etiquetas presentes no código fonte - como é o caso das aplicação de *web crawling*, minimizando a extração e análise que são processos que consomem elevados recursos da máquina.

No *text mining*, o sistema tem de interpretar a linguagem humana, encontrada na forma de texto não estruturado. De um modo geral, o algoritmo tem de ter uma definição para cada palavra como, se exprime um sentimento negativo ou positivo, por exemplo. A construção frásica utiliza algumas classes gramaticais - como, por exemplo, determinantes, advérbios, entre outras -, que na maioria das vezes não acrescentam valor para a perceção do contexto

[1] http://eaecnet.com/

abordado. Por outro lado, a sequência de palavras, que pode ter inúmeras combinações, tem um impacto enorme podendo alterar o sentido ao texto, continua a ser de difícil compreensão por parte dos algoritmos. O sistema pesquisa por novos termos no texto e procura relacioná-los com os termos conhecidos. No entanto, o sentido de um frase não pode ser interpretado meramente com base nas palavras usadas, mas também pelo encandeamento das mesmas.

1.2 Objetivos e resultados esperados

O presente projeto pretende reduzir o tempo de esforço na realização do boletim mensal da EAEC. Tendo os utilizadores um diretório com uma seleção de notícias de interesse, não existe a necessidade de despender demasiado tempo à procura de notícias. Ao fazer-se uso da capacidade computacional, a capacidade de processar quantidades substanciais de informação e disponibilizada em diversos lugares é potencializada.

Para tal propomos a seguinte estratégia:

- Seleção da tipologia dos *websites* e temas que são de interesse para o boletim da EAEC;
- Escolha e configuração de *software* para procura e recolha de artigos;
- Desenvolvimento de algoritmos de *text mining* que analisem as notícias recolhidas;
- Criação de um processo que avalie os resultados anteriores, aceitando ou rejeitando;
- Criação de um diretório acessível aos membros da EAEC, com sistema de permissões e organizado por categorias, para receção das notícias resultantes de todo o processo.

1.3 Estrutura do Documento

O presente documento começa com uma introdução, onde é feita uma breve descrição do contexto, do problema e da estratégia a seguir para uma necessidade que emergiu numa organização.

Logo de seguida, o Capítulo 2 Estado da Arte apresenta alguns dos conceitos e técnicas das tecnologias que serão abrangidas, bem como algumas soluções para fins semelhantes que já existem no mercado.

No Capítulo 3 a solução para o problema é analisada com maior detalhe e é feita uma descrição da arquitetura da solução a desenvolver.

As etapas da implementação são apresentadas no Capítulo 4 com a apresentação do ambiente onde o sistema foi desenvolvido, os processos para a obtenção de dados, o processamento dos mesmo em diferentes cenários e por fim, a estrutura para a apresentação para o utilizador final.

No Capítulo 5 Resultados são expostos e analisados os comportamentos dos processos criados quando testados com um conjunto de dados conhecido. A interface criada para o utilizador final é revelada e descrita.

Por fim, o último capítulo é composto por um resumo da solução implementada, os seus benefícios e algumas sugestões de melhorias futuras.

Capítulo 2

Estado da Arte

2.1 Processos de Text Mining

O encandeamento de algumas tarefas descrevem o processo de *text mining* para a classificação de texto. Na Figura 2.1, o diagrama apresenta de forma breve o momento desde a análise do documento até à previsão de uma classificação para o conteúdo encontrado, passando pela transformação dos dados em informações de maior valor que são usadas no treino de um modelo de classificador.

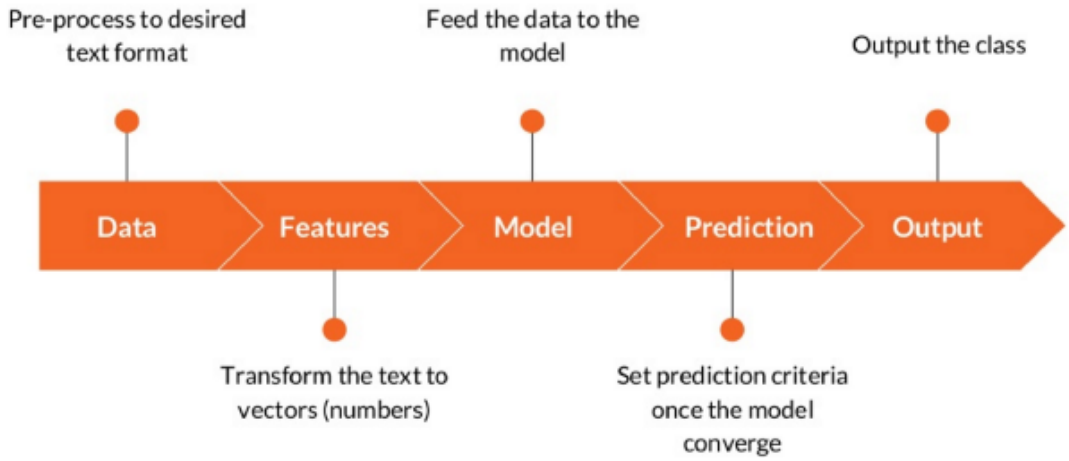

Figura 2.1: Diagrama do processo geral de classificação de texto em *text mining* {Fonte: Santos 2019}

2.1.1 Pré-Processamento do Texto

No *text mining*, os dados são pré-processados para a extração de termos com valor e conhecimento de dados não estruturados. Um bom resultado facilita as etapas seguintes do processamento dos dados, a redução de custos operacionais e produção de melhores resultados. O processo para ser completo divide-se em várias etapas.

2.1.1.1 Tokenização

A tokenização consiste na tarefa de separar palavras, números e sinais de pontuação em pequenos fragmentos chamados *tokens*. Por norma essa separação utiliza como delimitador

o espaço, com exceção de algumas línguas, como o chinês ou japonês (Indurkhya & Damerau, 2010). No entanto, existem outros casos particulares, como sinais de pontuação que aparecem junto das palavras (Jackson & Moulinier, 2002) e que normalmente até são dispensados por não acrescentarem valor. Por exemplo, na frase "Está tanto calor aqui!", aplicando este método obtêm-se cinco *tokens*: [Está], [tanto], [calor], [aqui] e [!].

2.1.1.2 Remoção de Stop Words

As *stop words*, ou palavras vazias como alguns autores sugerem como tradução, são termos que têm pouco valor para a análise, como por exemplo: preposições, conjunções, pronomes e determinantes. Os algoritmos de *text mining* recorrem a listas com as *stop words*, procurando-as nos dados em análise e removendo-as, reduzindo desta forma o tempo de processamento.

2.1.1.3 Stemming

No método de *stemming* são utilizados processos heurísticos na procura da raiz da palavra. Por exemplo, para as palavras florido, floral, flores e florescer, a sua raiz é flor. Com isto, é mais acessível fazer correspondência com as palavras utilizadas posteriormente na procura. Em termos da alocação de recursos, há uma poupança de espaço na memória, redução no número de palavras e regras a serem criadas. A redução de índices criados para as palavras pode ser reduzido entre 40 a 50% (Gurusamy e Professor n.d.). Um dos cuidados a ter em consideração é que algumas palavras apesar de parecerem derivadas a partir da mesma, têm significado diferente, por isso devem ser mantidas em separado. Para *stemming* existem disponíveis algumas bibliotecas conhecidas como a Porter stemmer, Lancaster stemmer, e Snowball stemmer (Porter, 1980).

2.1.1.4 Lemmatization

A *lemmatization* é uma tarefa que faz a análise morfológica das palavras de maneira a encontrar o infinitivo dos verbos ou uma única forma para os substantivos. Com isto, obtém-se uma melhoria na precisão e eficácia da análise posterior. Por exemplo as palavras giro, bonito e jeitoso, que podem ser transformados na palavra bonito (Plisson, Lavrac e Mladenić 2004).

2.1.2 Transformação de Texto

A transformação de texto é uma das técnicas de *text mining* mais importantes, sendo responsável pela seleção e utilização dos dados. Um dos resultados mais visíveis é a redução da dimensão do conjunto do dados (Iyyer et al. 2015).

2.1.2.1 Bag of Words

Bag of words, ou em português saco de palavras, é uma representação de texto que faz o registo do número de ocorrências que cada palavra tem num texto, não tendo em conta

a ordem em que estas surgem. Para tal, é necessário um vocabulário com as palavras conhecidas e uma medida para indicar a presença das palavras conhecidas. Tratando-se de um modelo simplista, que compara os documentos simplesmente pelo número de vezes que as palavras aparecem. Em termos de processamento não é muito pesado (Aryal et al. 2019).

2.1.2.2 Vector Spaces

O modelo baseado em espaços vetoriais de uma maneira grosseira pode dizer-se que é uma extensão do *bag of words*, pois regista também a frequência com que as palavras surgem. No entanto, a sua ordem neste modelo tem interesse.

O texto pode ser transportado para vetores que representam de forma binária (0,1) a presença dos termos ou é representado por varáveis atribuem um peso a cada termo.

Os documentos grandes apresentam poucas similaridades. A existência de palavras não relacionadas tem um elevado impacto no resultado, assim como termos semelhantes que não são usados na pesquisa. A utilização de métodos de análise com base no peso das palavras apesar de parecer intuitivo discrimina as ligações entre conceitos o que pode conduzir a resultados falsos (Jing, Huang e Shi 2002).

2.1.3 Classificação De Texto

A classificação de texto é o processo de atribuição de tags ou categorias ao texto de acordo com o seu conteúdo. Existem muitos modelos de machine learning para que podem ser usados para treinar o modelo final (Ikonomakis et al. n.d.).

2.1.3.1 K-Nearest Neighbor

O algoritmo *K-Nearest Neighbor* incide sobre os vetores com as variáveis do texto do documento. Um valor parametrizado define a distância a que certos termos estão próximos uns dos outros. Os vizinhos dos termos por sua vez também são classificados com essa base. Ao ser utilizado um grande número de conjunto de dados de treino, uma maior quantidade de palavras adjacentes vão ser consideradas como pertencentes a essa categoria, logo maior será o risco de má classificação. Além disso quanto maior for a distância de termos definida, maior será o impacto no desempenho (Bolandraftar, Bafandeh e And n.d.)(Banu e Chitra 2015).

2.1.3.2 Naive Bayesian

Os métodos de *Naive Bayesian* numa vista computacional são eficazes e apresentam um bom desempenho a prever, principalmente devido ao facto de processarem vetores de informação binário, independentemente da sua ordem. Nos textos encontram-se muitos termos, pelo que este algoritmo foca-se nos principais, reduzindo o ruído provocado por outras palavras e aumentando assim a precisão da classificação.
O *Naive Bayesian* encontra-se popularmente distribuído em dois modelos: o modelo de eventos multi-variado de Bernoulli e o modelo de eventos multi-nominal. Em ambos os modelos a probabilidade de cada classe é calculada igualmente para cada classe, no entanto

a probabilidade da classe a que cada termo pertence é diferente. No modelo multi-variado de Bernoulli, calcula-se quantas vezes uma palavra presente no *bag of words* surge. Já no modelo multi-nominal são apresentadas as palavras que existem no documento e a sua frequência, sem a criação de um*bag of words* com os termos da pesquisa, e seguindo o princípio de *Naive Bayesian* que enuncia que a aparição de cada palavra é independente das outras (Chen et al. 2009).

2.1.3.3 Support Vector Machines

O *Support Vector Machines* (SVM) (Wen et al. 2010) é um método de aprendizagem baseado no princípio da Minimização de Risco Estrutural, que aponta para a minimização de erros verdadeiros. Um dos princípios que mantêm este algoritmo é elevar os limites superiores da dimensão usada na análise de texto. O SVM tem ainda a capacidade de medir se está a ser usado um termo na sua forma mais primitiva ou se está a ser utilizado uma prefixação e/ou sufixação do mesmo.
Os algoritmos de classificação de textos na aprendizagem trabalham com muitos termos, tendo em alguns casos certas limitações. O SVM tem a capacidade de fazer um auto-ajuste, conseguindo contornar essa limitação permitindo assim trabalhar com um maior número de casos.

Os vetores dos documentos são dispersos. Alguns documentos só contêm algumas correspondências. Ao serem utilizados outros algoritmos esses têm a tendência a só utilizar termos vão aparecendo mais vezes. Os SVM por utilizar vetores densos e com instâncias dispersas, torna-se numa solução mais polivalente (Joachims n.d.).

2.1.4 Áreas de text mining

O *text mining* é o processo de procura e extração de conhecimento de informação útil a partir de dados textuais. A interpretação da linguagem humana por parte das máquinas é uma área de pesquisa que procura a descoberta de conhecimento de dados não estruturados. Sendo a *Internet* o repositório de notícias mais acessível e abrangente dos dias de hoje, torna-se no lugar mais promissor para a consulta de documentos S. Vijayarani, J. Ilamathi e Nithya 2015.

2.1.4.1 Information Retrieval (IR)

Information Retrieval (IR) é a área relacionada com os métodos de recolha de informações relevantes e associadas com os termos de pesquisa. Exemplo disso, é o que é elaborado pelo motor de pesquisa Google, para determinadas palavras, ele devolve um conjunto de websites com a informação mais pertinente. De outras formas menos visíveis, o rasto deixado pelo utilizador durante a navegação pelas plataformas digitais, acaba por ser utilizado para devolver outros conteúdos, como publicidade direcionada aos interesses do mesmo (Talib et al. 2016a).

2.1.4.2 Information Extraction (IE)

A área de *Information Extraction* (IE) é responsável pela extração de informação de um documento ou conjunto de documentos de relevo previamente recolhidos. A informação do documento está presente sob a forma de dados não estruturados e, de modo, a serem aplicados algoritmos de classificação, necessita de ser transformada em informação estruturada. No *text mining* procura-se a descoberta em geral, de dados não conhecidos e novas relações, o IE procura extrair informações específicas e estruturas e relações conhecidas. Uma das abordagens que o IE utiliza é a criação de regras de correspondência de padrões para identificar as entidades desejadas e as relações. No entanto, este método é complexo e o sistema necessita de reajustamento caso novas variáveis sejam acrescentadas. Outro método é a utilização de sistemas de aprendizagem supervisionados, que criam padrões com base num conjunto de dados de treino, e sob a supervisão do ser humano, um conjunto de padrões. Esses padrões podem ser feitos através da etiquetagem de cada *token* e utilizando de seguida modelos de sequência estatística. De uma maneira geral, um sistema de classificação consegue prever qual será a etiqueta a atribuir a palavras caso as palavras próximas já estejam identificadas (Mulins e Mullins 2008) (Nahm e Mooney 2002).

2.1.4.3 Natural Language Processing (NLP)

O *Natural Language Processing* foi desenvolvido para o ser humano ser capaz de comunicar com o computador sem necessitar de saber a linguagem dele. A maioria dos algoritmos e bibliotecas existentes estão desenvolvidos para trabalhar com a língua inglesa, por isso, a tradução do texto, numa primeira fase, quando se pretende trabalhar noutros idiomas, é a melhor abordagem. O NLP tem em consideração algumas das seguintes terminologias:

2.1.4.3.1 Morfologia Um morfema é a forma mais redutora que uma palavra pode ter de modo a ser identificada e ter significado. Os radicais das palavras podem ter prefixo, sufixos ou ambos. A utilização de morfemas no desenvolvimento e no processo dos algoritmos leva a uma maior simplicidade nas regras construídas à volta do mesmo, bem como no processamento.

2.1.4.3.2 Léxico O léxico corresponde ao significado das palavras. O NPL atribui uma *tag* a cada palavra, sendo que posteriormente uma ligação entre os diferentes termos pode ser feita, de modo a perceber o significado semântico do contexto do texto.

2.1.4.3.3 Sintaxe A sintaxe realiza uma análise da frase para entender a sua estrutura gramatical. Cada palavra tem uma função nas frases e dependem de outras para ter coerência, gerando uma ação realizada. A ordem das palavras é relevante, bem como se estão após a um predicado ou um nome, por exemplo.

2.1.4.3.4 Semântica Certas palavras têm mais que um significado. Ao dizer-se "tens a manga do casaco suja"ou "aquela manga está tão saborosa", a palavra "manga"na primeira frase refere-se a uma parte do vestuário e na segunda refere-se a um fruto. Apenas fazendo a análise completa da frase permite saber o significado de alguns termos.

2.1.4.3.5 Discurso A análise de texto por frase pode ser inconclusiva porque existe informação que é explicitada nas sentenças próximas. O ser humano tem por hábito, por exemplo, utilizar pronomes para após referir a quem/ao que se estava a referenciar (Resolução Anáfora). A estrutura do texto também pode adicionar significado ao texto.

2.1.4.3.6 Pragmática A utilização de alguma linguagem dentro de certas situações pode ocultar maior significado que em termos leigos se pode percecionar. Compreender as intenções, planos e objetivos requer conhecimentos gerais. O processo de produzir frases, declarações ou parágrafos a partir de representação internas de conhecimento é chamado de *Natural Language Generation* (NLG). O procedimento consiste em identificar os objetivos, planear como os mesmos devem ser atingidos (avaliando a situação e as fontes de comunicação) e realizar os planos em forma de texto (Talib et al. 2016b).

2.1.4.4 Web Mining

O Web Mining consiste numa técnica que encontra ou extrai informação relevante de páginas *web*. Relaciona-se com o *text mining* devido ao fato da maioria do conteúdo estar em texto não estruturado, no entanto obtém uma maior vantagem da semi-estruturação natural das páginas *web* (Cooley, Mobasher e Srivastava 1997).
As tecnologias ao dispor incluem o *Natural Language Processing* e a *Information Retrieval*. A ferramenta tem capacidade de analisar as ligações existentes no *website* de modo a mapear a estrutura do mesmo, utilizando igualmente informações provenientes do formato HTML (*Hyper Text Markup Language*) ou do XML (*Extensible Markup Language*). O resultado pode permitir classificar o site quanto à sua importância na *Internet*, bem como das suas páginas (algumas têm mais relevância, em relação a outras que têm uma utilidade menos expressiva).
Na área de *Web Mining* existe uma vertente que realiza uma análise do padrão de utilização por parte dos navegadores. O principal objetivo é prever o comportamento do utilizador aquando da sua interação com o *website*, com base na sua utilização ou na de outros semelhantes, mesmo tendo sido esta feita em outros servidores *web*. As três fases envolvidas são o pré-processamento, descoberta de padrões e análise de padrões.

2.2 Soluções Existentes

Nesta secção pretende-se apresentar algumas das soluções mais conhecidas, encontradas no mercado.
Na Tabela 2.1 encontra-se a lista dos softwares mais popular e as funcionalidades associadas à área de *text mining* que cada produto oferece. Os *softwares* são responsáveis pela extração de informação, que a par da categorização são as funcionalidades mais populares, sendo oferecidas por todos os produtos. Na tabela 2.2, estão apresentadas as áreas de negócio que mais procuram estas tecnologias: em medicina o uso é popular para a ligação entre substâncias ou sintomas ou para organização; na educação para o resumo das pesquisas; nos negócios para a visualização de padrões nas tendências dos consumidores; e no governo para a deteção de irregularidades (Gooch 2011a).

Tabela 2.1: Produtos e Tecnologias de *text mining*

Funcionalidades contidas em alguns dos principais produtos do mercado

	Smart Discovery	SAS Text Miner	TextAnalyst	Watson Natural Language Understanding
Extração de Informação	X	X	X	X
Resumo	X		X	X
Categorização	X	X	X	X
Ligação de Conceitos		X		
Clustering			X	X
Visualização de informações	X			

Tabela 2.2: Áreas de Negócio e Tecnologias de *text mining*

Tecnologias de *text mining* mais usadas em algumas Áreas de Negócio

Área de Negócio	Extração de Informação	Resumo	Categorização	*Clustering*	Ligação de Conceitos	Visualização informações
Medicina						
FAQ	X		X		X	
Desenvolvimento de Fármacos	X			X	X	
Novos Tratamentos					X	
Negócio						
Análise Competitiva		X				
Infração da Propriedade Intelectual	X			X		
Deteção nas Redes Sociais						X
Personalização de Conteúdo				X		
Governo						
Deteção de redes terroristas	X			X	X	X
Prevenção e deteção de Crime	X			X	X	X
Educação						
Pesquisa de um Tópico		X	X			
Análise de citações	X			X		X

2.2.1 Smart Discovery

Smart Discovery[1] atualmente é uma ferramenta do SAP Analytics Cloud, usado para explorar dados e procurar informações valiosas.
Algumas das funcionalidades que o software oferece são:

- Descoberta dos principais indicadores que influenciam os KPI (*Key Performance Indicator*), bem como informações sobre os mesmos;

[1] https://www.sapanalytics.cloud/resources-smart-discovery-update/

- Identificação de *outliers*;
- Análise de padrões;
- Uso de histórico para prever futuros resultados;
- Simulação de cenários 'e-se'.

Smart Discovery permite às empresas carregar um conjunto de dados para a plataforma de forma a serem analisados. A ferramenta realiza uma pré-análise em que retoma algumas das dimensões e medidas que podem ser utilizadas na análise dos itens de interesse.
À medida que a análise evolui, o próprio programa assimila dados e sugere previsões. O utilizador alterando os fatores chave e filtrando os dados, alcança dados mais relacionados com certas áreas de interesse. A previsão permite detetar situações fora do esperado, como um fator que não parecia importante, mas que irá retardar o crescimento do negócio.
A ferramenta permite às empresas a simulação de cenários hipotéticos. A utilização de certas medidas e a manipulação de outras permite ver os impactos dos resultados, levando as organizações a implementarem certas mudanças (Gooch 2011b).

2.2.1.1 SAS Text Miner

A SAS, uma empresa que dedicada à análise, produziu a ferramenta SAS Text Miner[2] para análise de dados não estruturados presentes na web, comentários, livros ou de outras fontes textuais.
O objetivo consiste em melhorar o desempenho do modelo de previsão através da utilização de dados não estruturados. A maioria das previsões feitas no mercado recorrem a algoritmos de *data mining* que trabalham com dados estruturados, estes por não disporem de tantas informações, pode não ser considerado conhecimento importante; automatizar atividades manuais demoradas, como a extração de temas ou relacionamento de termos-chave, usando técnicas de NLP e *Machine Learning* (ML); permitir fornecer uma lista personalizada, numa interface agradável, aos algoritmos de *text mining* para refinação das regras e tópicos criados; fazer uma ponte entre uma interface gráfica agradável ao utilizador e as representações numéricas utilizadas nos modelos de análise de texto. As funcionalidades que a SAS anuncia para o programa são:

- Elevado desempenho na análise de grandes quantidades de texto;
- Interface flexível e *user-friendly*;
- Criação de regras booleanas automaticamente para uma fácil classificação dos textos;
- Avaliação da relevância dos termos e do seu uso ao longo dos tempos;
- Identificação dos temas presentes nos documentos;
- Permitir adicionar entidades personalizadas;
- Fácil importação dos textos;
- Suporta vários idiomas (*Text Mining Software, SAS Text Miner | SAS* 2020).

[2] https://www.sas.com/pt_pt/software/text − miner.html

2.2.1.2 TextAnalyst

O TextAnalytics[3] é um produto que resulta da combinação de esforços entre a Megaputer Intelligence e a MicroSystems, para uma ferramenta inteligente para análise da semântica, sumarização e navegação em textos escritos na linguagem natural.
O produto promete reduzir drasticamente o tempo que um analista demora a compreender documentos em que tem conhecimentos insuficientes sobre os temas abordado, o que resulta numa compreensão mais profunda. A realização de reduções e análises preliminares de dados simultaneamente, sem perda de informações é um outro grande atributo do software.
Algumas das tarefas que o TextAnalytics realiza completamente de forma automática são:

- Criação de uma rede semântica para os textos analisados;
- Resumo de documentos;
- Formação de uma base de dados de conhecimento;
- Pesquisa semântica por informações usando uma método de criação de uma sub-árvore de conceitos;
- Estruturação dos tópicos de forma hierárquica;
- Indexação de texto (*TextAnalyst - new text mining solution from Megaputer - Megaputer Intelligence* 2020).

2.2.1.3 IBM Watson Natural Language Understanding

A IBM Watson Natural Language Understanding[4] é um microserviço que opera com *cognitive systems*, um conceito vanguardista que responde à necessidade de navegar pela constante corrente de dados não estruturados de forma eficaz. Um dos princípios assenta na utilização do maior número de informações disponíveis para alcançar resultados precisos. A IBM explica que o software não se concentra no significado dos termos, mas sim de recursos linguísticos que são utilizados pelas pessoas, ou seja, para uma passagem de texto (a que a IBM chama questão) infere outra passagem de texto (a que a IBM chama resposta) (High n.d.). Os benefícios do software que a IBM dá mais destaque são:

- Deteção de entidades como pessoas, lugares e eventos;
- Categorização dos dados em diferentes níveis de granularidade;
- Identificação de conceitos que podem não estar diretamente referenciados no contexto;
- Extração de emoções/sentimentos de frases ou do documento como um todo;
- Identificação de relações entre termos;
- Extração rápida de metadados como o autor, título e data de publicação do documento;
- Identificação dos papéis semânticos nas sentenças como o sujeito, a ação e o objeto (*Watson Natural Language Understanding - Features | IBM* 2020).

[3] https://www.megaputer.com/solutions/text-analytics/
[4] https://www.ibm.com/cloud/watson-natural-language-understanding

Capítulo 3

Análise e Desenho

Neste capítulo é descrita a solução numa perspetiva mais tecnológica e apresentados todos os detalhes da arquitetura da solução a implementar. Na primeira secção é compreendido o problema e são apresentadas as categorias a serem utilizadas para as informações publicadas na World Wide Web (WWW) no contexto do programa ERASMUS+, visto estas serem a base dos interesses da European Association of ERASMUS Coordinators. Na segunda secção são listadas as fontes de informação de onde o conteúdo será extraído. De seguida, a classificação dos documentos é clarificada com descrição dos parâmetros de seleção.

Na última secção, um desenho da arquitetura é elaborado com a divisão do sistema em componentes, dadas as funções e tecnologias distintas empregues. As funcionalidades são descritas e desenhado um esboço dos processos a serem desenvolvidos.

3.1 Categorias

A EAEC para a produção do seu *newsletter* mensal faz uma exploração dos conteúdos publicados no últimos mês em sites de interesse. O processo de pesquisa e análise de informação é feito de forma manual pelos seus colaboradores. A EAEC tem a vontade de automatizar a recolha de informação e disponibilizá-la em diretorias associadas a um tópico. A diminuição do tempo despendido na recolha de notícias, permitirá uma redução significativa na duração da produção do *newsletter*.

Os interesses da European Association of Erasmus Coordinators concentram-se na:

- troca de informação e experiência entre os membros e afins;
- promoção da mobilidade dos estudantes, académicos e administrativos no território da União Europeia;
- circulação das tendências no Processo de Bolonha e as suas atualizações nos círculos académicos;
- promoção dos princípios do projeto ERASMUS+ no Programa da Comissão Europeia;
- elevação nos padrões de educação e qualidade no território da União Europeia;
- organização de seminários e conferências, assumindo um papel para a comunicação direta entre os membros da associação e catalisando o aumento da mobilidade por toda a Europa e afins, bem como definir novas áreas nas atividades da associação;

- preparação e submissão de propostas de projetos para serem financiadas pela Comissão Europeia;
- cooperação como parceiro em projetos europeus financiados;
- inquirição e estudos nos tópicos abordados em cima;
- produção de publicações de interesse para os membros (*EAEC Network - About EAEC* 2020).

Desta feita, podem-se agrupar os conteúdos de relevo para a EAEC nas seguintes categorias:

3.1.1 Eventos

Um evento é qualquer acontecimento previamente planeado, organizado e coordenado com o intuito de reunir o maior número de pessoas em um mesmo espaço físico e temporal, com informações, medidas e projetos sobre uma ideia, ação ou produto, apresentando os diagnósticos de resultados e meios mais eficazes para se atingir determinado objetivo (Martin 2003).

3.1.2 Projetos

Um projeto é um esforço temporário que tem como propósito um resultado único e possui recursos delimitados. Por norma, existem datas de início e fim definidas; a intenção de criar um produto ou serviço, ou melhorar algo já existente; e uma estipulação inicial dos custos e recursos, para que não se exceda a verba disponível e não faltem recursos para a sua conclusão. Um projeto pode ser cultural, empresarial, de pesquisa, pessoal ou social. Justo 2018

3.1.3 Mobilidade

As atividades de mobilidade pretendem proporcionar aos cidadãos os meios necessários para participarem ativamente no mercado de trabalho e sociedade em geral. Neste sentido, o programa Erasmus+ promove atividades para a compreensão e participação na sociedade - tanto no país materno como com outras culturas e povos -, para um aumento das competências, empregabilidade e consciência cultural (*Projetos de mobilidade nos domínios da educação, formação e juventude | Erasmus+* 2020).

3.1.4 Políticas e Recomendações

As organizações participantes contribuem com medidas para a melhorar a qualidade dos sistemas em matéria de ensino, formação e juventude na Europa, a promoção da aprendizagem transnacional e a cooperação entre autoridades (*Apoio à reforma das políticas | Erasmus+* 2020).

3.1.5 Outros

Todos os assuntos não abrangidos pelos tópicos descritos acima. Na maioria dos casos, as páginas web classificadas como "Outros", dizem respeito a partes do *website* que não contêm propriamente notícias, como páginas de autentificação, índices, FAQs, contactos, entre outros.

3.2 Proposta de Desenvolvimento

O projeto devido à utilização de diversas tecnologias e tarefas distintas, tem de ser tratado em diversos componentes. Um diagrama do sistema pretendido é apresentado na Figura 3.1.

Uma componente do processo proposto é a pesquisa e *download* de páginas *web*. A procura de notícias deve ser feita com uma frequência que permita um número significativo de novas notícias. A chamada do *script* de *web crawling* deve procurar informação a partir dos URL indicados pelo utilizador como *seeds*, incluindo hiperligações para onde será direcionada a pesquisa e a informação analisada e assim recursivamente. A informação dos endereços *web* explorados tem de ser armazenada na base de dados com a indicação da hora da recolha. Os registos na base de dados permitirão conhecer o crescimento do volume de páginas no *website* e ajustar os ciclos de *crawling* para cada *seed* URL. Outro ponto a ter em conta na recolha de um conjunto total de páginas, a partir do mesmo local, em alturas diferentes, será que um grande número de páginas já foi previamente descarregada, e posteriormente processada, pelo que será necessário um mecanismo de eliminação de redundâncias.

Os ficheiros resultantes do processo de *web crawling* encontra-se em formato HTML, com elementos que são característicos dessa estrutura, mas que em nada acrescentam informação para a classificação do tópico do documento, acabando somente por prejudicar a análise. A remoção das *tags* deve acontecer antes da análise ao corpo da notícia.

O ser humano utiliza um vocabulário com palavras conjugadas, derivadas e sinónimas de termos mais comuns, bem como palavras que servem somente de ligação, como advérbios. Torna-se importante implementar funções para a redução da disparidade de termos similares e ordena-los em vetores, para melhorar o desempenho e compreensão feita por parte dos algoritmos de *machine learning*.

Um classificador deve ser empregue sobre os os documentos de texto e aplicada uma previsão da categoria para cada um. As previsões são armazenadas na base de dados. O conjunto de treino é relativamente pequeno, pelo que o classificador deve ser recriado ao longo de cada ciclo e à medida que novos documentos recebem a garantia por parte do utilizar da classificação. Mais detalhes deste processo são apresentados na Secção 3.2.1.

Por fim, a informação deve ser possível de ser consultada e/ou manipulada numa plataforma pelos utilizador. Na Secção 3.2.2 podem-se encontrar mais detalhes deste componente.

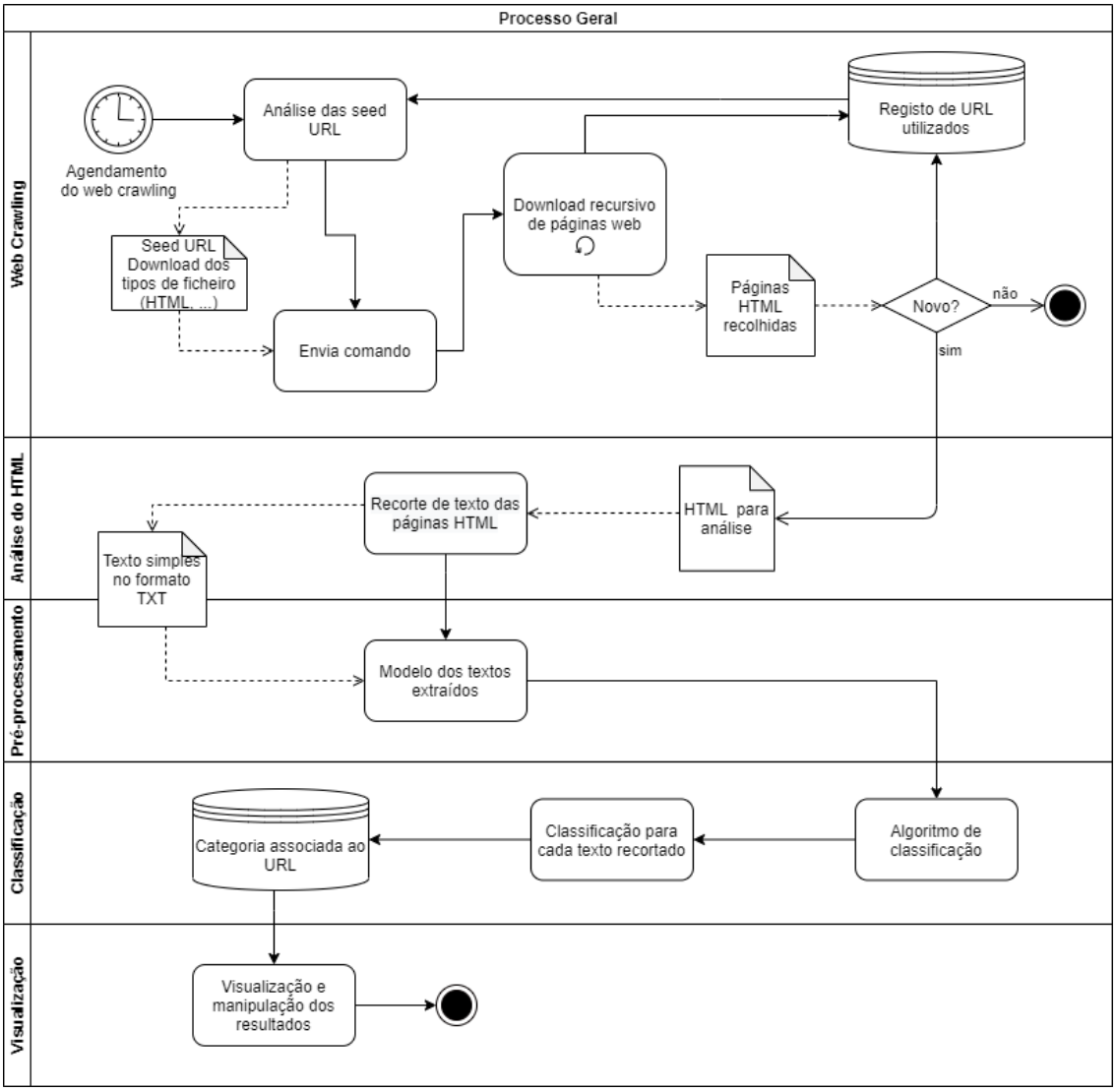

Figura 3.1: Visão geral do processo de recolha e classificação.

3.2.1 Classificador

Um classificador é um algoritmo que analisa os dados seguindo uma metodologia própria e recorre a um conjunto de características para analisar um corpo de texto, a fim de devolver o tópico abordado pelo mesmo, conforme pode ser observado na Figura 3.2.

Numa primeira fase é necessário treinar as características que o classificador utiliza como associação a determinado tópico. Para tal, é preciso definir para um conjunto de dados alargado o respetivo tópico, de forma manual, o chamado conjunto de dados de treino. Depois de um pré-processamento dos dados, o algoritmo de classificação é empregue sobre os dados e são aplicadas técnicas de validação dos parâmetros que devolvem melhores resultados. De frisar, que nesta fase o conjunto de dados de treino inicial é dividido em duas partes: conjunto de dados de treino e conjunto de dados de teste.

Com o modelo de classificador treinado após a leitura de documentos resulta a previsão de um tópico associado. A informação deve ser armazenada na base de dados para posterior uso.

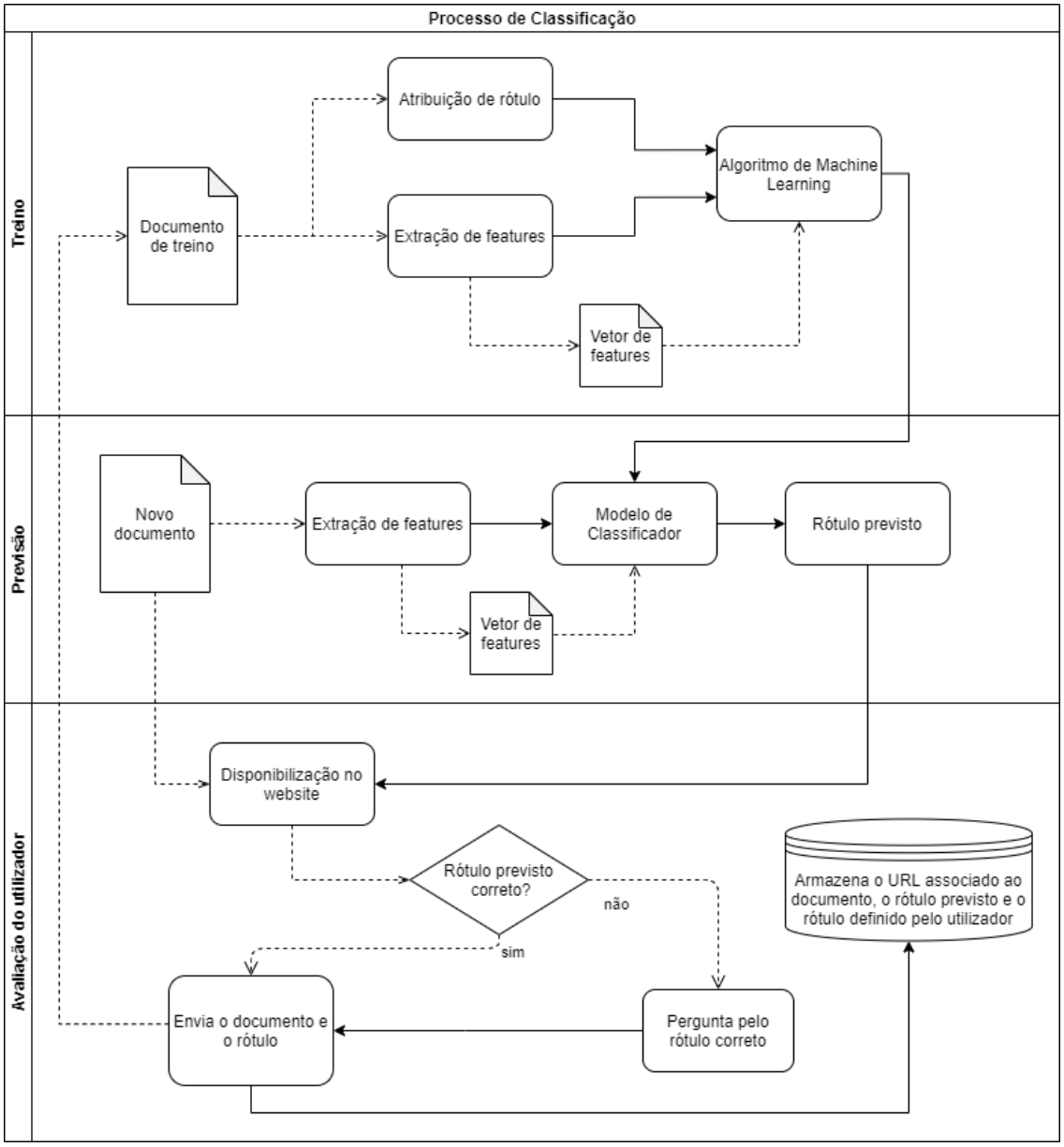

Figura 3.2: Visão geral do processo de classificação.

O utilizador ao consultar os dados deve ter a opção de validar a classificação prevista (confirmar ou alterar para a correta). Sendo dados confiáveis, e sendo o conjunto de dados de treino pequeno, devem ser usados para melhorar o sistema de classificação - trata-se de um sistema de aprendizagem ativo. Os novos documentos validados pelo utilizador passam a fazer parte do conjunto de dados de treino e o modelo de classificador é novamente treinado e as classificações revistas, esperando-se assim um aumenta na qualidade das classificações.

3.2.2 Ferramenta de Visualização

A interação entre o sistema e o utilizador faz-se através dum *website*. Não só para ver a previsão feita pelo classificar, mas para um conjunto de outras tarefas.

A apresentação de todos os documentos recolhidos deve conter o URL de origem e o documento em texto simples. A par do registo deve estar a classificação prevista pelo classificador desenvolvido. O utilizador deverá ter a opção de confirmar a categoria sugerida ou escolher de uma lista de possibilidades a categoria mais adequada.

Para a publicação do *newsletter*, os membros da EAEC têm que selecionar algumas das notícias recolhidas. Neste sentido, o utilizar deve estar registado e com sessão iniciada deve puder adicionar documentos a um repositório específico. Quando não precisar da notícia também deve ser permitido eliminar o registo do repositório.

Por fim, um módulo com dados estatísticos deve existir. O número de novos documentos, a percentagem de classificações corretas ou validadas, são alguns dos indicadores que podem ser úteis para análise do sistema.

3.2.2.1 Filtros e Palavras-chave

O número de documentos no primeiro *crawling* vai ser elevado e nos próximos irá crescer ainda mais. Uma procura numa lista de artigos sem apoio é uma tarefa impraticável. Desta feita, a plataforma *web* deve apresentar filtros que possam ser aplicados para os conjuntos de dados mais relevantes, como filtragem por site e categoria, por exemplo.

Nos moldes descritos acima, o utilizador está limitado à procura de notícias por categoria. A opção de procura de determinado tema em específico deve ser implementada. Através da pesquisa por palavras-chave devem ser devolvidos todos os documentos nos quais os termos se encontrem no corpo do texto, ao passo do que acontece com um motor de pesquisa.

3.2.3 Base de dados

O armazenamento de dados de forma permanente auxilia a gestão do sistema. Um conjunto de tabelas deve assegurar a informação sobre os documentos recolhidos, categorização, membros registados, repositórios e sementes do processo de *web crawling*. Deste modo, uma base de dados relacional torna-se uma boa solução para a representação dos dados e seus relacionamentos. Na Figura 3.3 encontra-se proposto o diagrama entidade-relacionamento (DER) com as tabelas:

Records Tabela responsável por guardar o URL e o nome do *website* associados à página web, o contexto do *crawling* (tópico) e as localizações no servidor local da página *web* originalmente extraído e do documento de texto resultante da extração das *tags* e outros elementos usados na estrutura *web*.

Records Crawler Tabela onde são registadas todas as vezes que determinado URL é descarregado pelo *web crawler*.

Categories Tabela que contem a informação da classificação prevista pelo classificador, bem como a retificada ou confirmada pelos utilizadores para cada documento.

List Categories Tabela que armazena todos os tópicos e categorias pré-definidos para serem usados na classificação.

Personal Repository Tabela onde ficam anotados os documentos que o utilizador guardou no seu repositório.

Users Tabela que contem o nome e as credenciais dos utilizadores registados.

Seed URL Tabela que contem as sementes usadas pelo processo de *web crawling* e o respetivo nome do *website*.

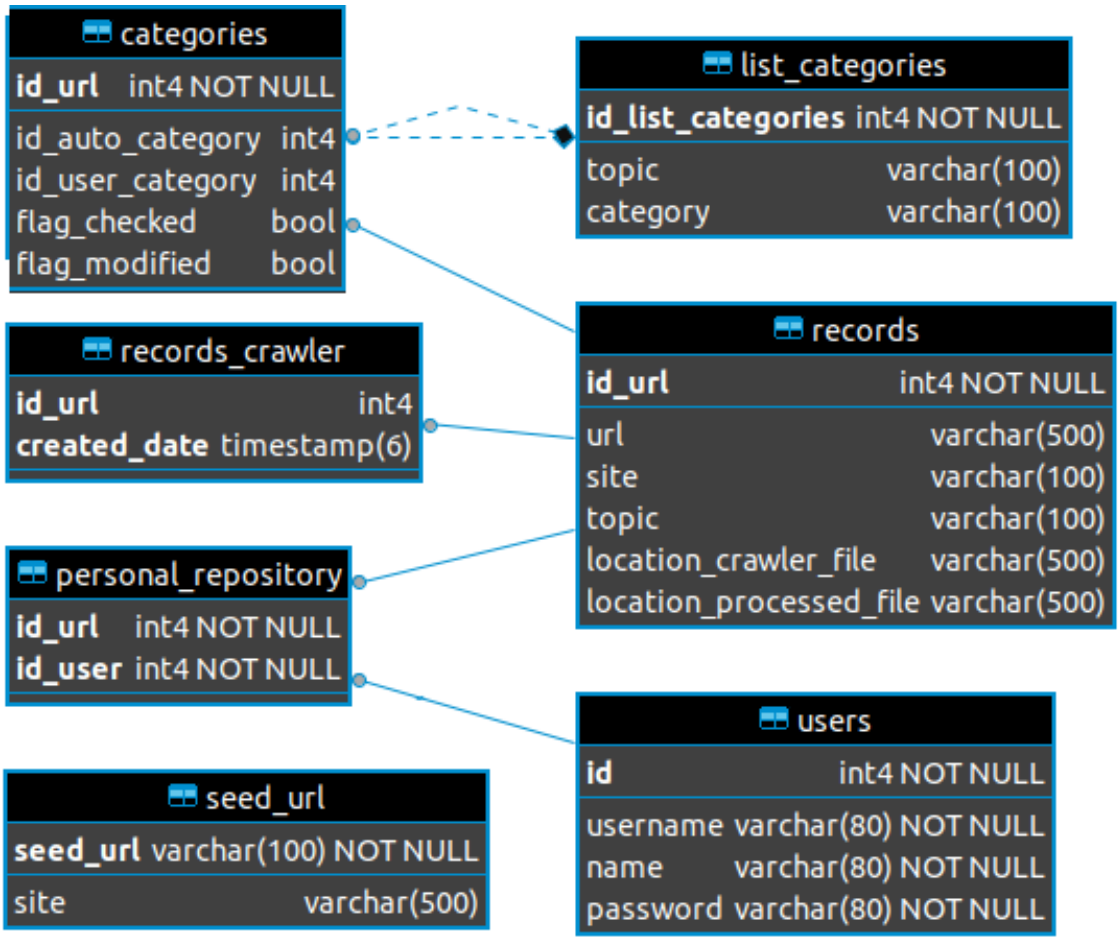

Figura 3.3: Diagrama entidade-relacionamento da base de dados.

3.3 Fontes de Informação

Na análise feita, os*websites*, ou partes de *websites*, que se identificaram com os objetivos da EAEC - e que são utilizados como *seed* URL durante a fase de *crawling]* - são:

- `https://eacea.ec.europa.eu/national-policies/eurydice/news_en`;

- `https://esn.org/news`;
- `https://www.erasmus-entrepreneurs.eu`;
- `https://www.esu-online.org/news`;
- `https://www.erasmuswop.org/news`;
- `https://eit.europa.eu/news-events/news`;
- `https://eacea.ec.europa.eu/erasmus-plus/news_en`.

A aplicação deve permitir a integração de novos URL, sem qualquer configuração adicional do sistema desenvolvido.

O conteúdo necessário da página *web*, para o processo de classificação, é o título e o corpo da notícia. Desta forma, elementos não textuais não devem ser considerados, como imagens, vídeos, áudio, entre outros. Além disso, folhas de estilo, *scripts* e *tags* usados para produzir a estrutura e conteúdo visual do *website* devem ser dispensadas.

3.4 Avaliação dos Modelos

Os modelos de classificadores depois de treinados são avaliados pelos resultados das suas previsões. Neste sentido, o conjunto de dados de treino inicial, constituído por um conjunto de documentos previamente rotulado com uma categoria, deve ser repartido em dois conjuntos de dados: treino e teste.

As técnicas de validação cruzadas Randomized Search e Grid Search devem usar o conjunto de dados de treino para a análise dos parâmetros dos modelos de classificação que produzem os melhores resultados para o tipo dos dados recolhidos, bem como para o ajuste do próprio modelo aos dados.

Com o modelo de classificador treinado devem ser feitos testes para avaliação da sua qualidade, em dados ainda não vistos pelo próprio. Os resultados da previsão do conjunto de dados de teste dão a indicação do comportamento do classificador. Através de fórmulas estatísticas devem ser avaliadas as discrepâncias entre a categoria prevista e a pré-definida. Os modelos com as menores taxas de erro na previsão significam maior confiança para o utilizador.

As métricas usadas na avaliação do classificador com melhores resultadas serão:

RMSE - Raiz Quadrada do Erro Médio Quadrado A RMSE é uma medida muito popular na avaliação de modelos de classificação. O resultado obtêm-se fazendo a diferença entre o valor previsto e o verdadeiro, elevar isso ao quadrado, fazer a média para todos os casos e por fim, aplicar a raiz quadrada, como se pode ver na Equação 3.1. Os seus valores podem variar de 0 até ao infinito, sendo que valores mais baixos significam erros menores, por isso são valores melhores. Penaliza grandes erros, pelo que os outliers têm um enorme impacto (Chai e Draxler 2014).

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}\left(\frac{d_i - f_i}{\sigma_i}\right)^2} \tag{3.1}$$

MAE - Erro Médio Absoluto O MAE é muito utilizado em problemas de regressão. O valor de erro é definido pela diferença entre o valor previsto e o verdadeiro, o valor absoluto é retirado, somado com os restantes e no final é feita média, como se pode ver Equação 3.2. O resultado é a média da grandeza de distanciamento entre o previsto e o actual.

$$MAE(y, \hat{y}) = \frac{1}{m} \sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{3.2}$$

Precisão A precisão refere-se a quão próximos estão os valores uns dos outros.

$$preciso = \frac{VP + FN}{VP + FP + FV + FN} \tag{3.3}$$

Exatidão A exatidão refere-se a quão perto do valor real estão os valores em questão.

$$exatido = \frac{VP}{VP + FP} \tag{3.4}$$

Recall O *recall* calcula da quantidade de valores de determinada classe, quantos desses valores foram bem previstos.

$$recall = \frac{VP}{VP + FN} \tag{3.5}$$

F1-score O F1-score é a média harmónica entre a precisão e o *recall* (Goutte e Gaussier 2005).

$$f1 - score = \frac{2 * preciso * recall}{preciso + recall} \tag{3.6}$$

Capítulo 4

Implementação

Neste capítulo começa-se por apresentar o ambiente onde o sistema é desenvolvido. De seguida, encontra-se descrito o processo de recolha de informação e disponibilização dos ficheiros. Aos documentos são retiradas todas as marcações das linguagens *web* e todos os termos não eliminados são indexados. Com todos os dados já disponíveis para serem analisados, um conjunto de modelos de classificação são treinados para a criação de um classificador capaz de classificar o assuntos das notícias recolhidas. Por fim, são apresentadas partes da interface desenvolvida para servir o utilizador na visualização e manipulação do sistema.

4.1 Ambiente de Desenvolvimento

O sistema operativo utilizado no desenvolvido foi o Ubuntu 20.04.1 LTS (Focal Fossa). A escolha recaiu pela sua estabilidade, facilidade de instalação de ambientes de desenvolvimento e o poder que o terminal proporciona. As facilidades e possibilidades de configuração que o Ubuntu oferecem facilitam e muito nas configurações de sistema.

A linguagem de programação preferencial na criação de *scripts* foi o Python 3.8.2. "O Python é uma linguagem fácil de aprender e poderosa. Ela tem estruturas de dados de alto nível eficientes e uma abordagem simples mas efetiva de programação orientada a objetos. A elegância de sintaxe e a tipagem dinâmica do Python aliadas com sua natureza interpretativa, o fazem a linguagem ideal para programas e desenvolvimento de aplicações rápidas em diversas áreas e na maioria das plataformas"(*O Tutorial Python — documentação Python 3.8.6* 2020).

PostgreSQL é um poderoso sistema de base de dados relacional de objetos *open-source* que usa e estende a linguagem SQL. Devido à arquitetura comprovada, confiabilidade, integração de dados e alto desempenho tornou-se uma opção válida para o presente projeto (O.A. 2016).

4.2 Web Crawling

A fase de *web crawling* é responsável pela procura e recolha de documentos pela World Wide Web para o servidor local. Nas subsecções seguintes são descritas as etapas adicionais necessárias para indicação dos pontos de partida, critérios de pesquisa e recolha, processamento do ficheiros descarregados e sua análise para armazenamento de metadados e deteção de redundâncias.

4.2.1 Ciclos de Crawling

A divulgação de notícias é feita com uma frequência diferente para cada *website*. Para uma maior eficiência do uso de recursos um *script* em Python avalia o número de novas publicações para cada *seed* URL. Os registos do histórico de descargas das páginas *web*, armazenados na base de dados, permitem calcular o crescimento de cada *website*.

A frequência do *web crawling* dos *seed* URL foi calculada usando o número de documentos recolhidos para os dias em que existiu um [crawling. Desta feita, no início de um ciclo de *web crawling* é calculada a média diária de documentos para todos os *seed* URL. A distribuição da frequência foi decidida da seguinte maneira: mais de três novas páginas, é diária; entre uma a três novas páginas, no início de cada quinzena; e igual a zero, no final do mês.

No caso dos *seed* URL adicionados recentemente pelo utilizador, como não existem ainda dados sobre a frequência de publicação de documentos para essas hiperligações, o seu *crawling* é diário nos primeiros dois dias. Ao terceiro dia, existindo já dados do número de notícias divulgadas em 24 horas, a frequência para este *seed* URL é definida pela primeira vez. Caso, por coincidência os primeiros dias tenham sido atípicos, no número habitual de publicações, como frequência de *web crawling* é calculada cada vez que existe um *crawling* a situação é alterada.

A função de cálculo da frequência de *crawling* é executada diariamente no início do *workflow* e gera a lista de *seed* URL a ser utilizada pelo *web crawler*.

4.2.2 Web Crawler

O *web crawler* escolhido, para a navegação pela World Wide Web e o descarregamento de páginas *web*, foi o GNU Wget, um *software open-source*. Por ser uma ferramenta de linha de comando não iterativa, pode ser facilmente chamada em *scripts* e em *cron tasks*, sendo assim facilmente incorporado num ambiente de integração contínua.

Um *script* foi criado com a chamada da ferramenta Wget e a invocação de algumas opções:

- '–recursive' - ativa a recursividade;
- '–level 2' - define o nivel 2 de profundidade máxima de recursividade;
- '–execute robots=off' - permite aceder a um número maior de páginas web;
- '–no-parent' - desativa o acesso aos diretórios ascendentes. Exemplo, em exemplo.pt/news, o crawler não acede a páginas fora do diretório de interesse: 'news';
- '–user-agent='Mozilla/5.0 (...)' - simula que o acesso está a ser feito por um navegador de internet, reduzindo assim possíveis bloqueios;
- '–random-wait' - adiciona um tempo aleatório entre pedidos para reduzir possíveis bloqueios;
- '–reject '*.js,*.css, (...)' - permite que certas extensões de ficheiros, que não são de interesse, não sejam descarregadas;
- '–ignore-tags=img,link,script' - ignora a procura recursiva de documentos para descarregar nestas tags HTML;

- '–header="Accept: text/html"' - especifíca que procura conteúdo do tipo texto ou HTML;
- '–follow-tags=a' - dita que as hiperligações usadas pela recursividade na procura de documentos devem aparecer na tag "a"do HTML;
- '–no-check-certificate' - ignora o facto do website poder não ter um certificado válido;
- '–show-progress' - apresenta mais informação sobre o processo de crawling;
- '–adjust-extension' - para casos em que a página web descarregada não tem extensão, esta opção atribui a extensão .html;
- '-a /home/(...)' - acrescenta todas as mensagens ao ficheiro de log;
- '–directory-prefix=/home/(...)' - define o local onde os ficheiros vão ser descarregados;
- '-i /home/(...)' - localização do ficheiro de texto com a informação dos seed URL.

4.2.3 Transformações nos Documentos Descarregados

Os documentos descarregados por vezes não foram suficientemente filtrados na ferramenta de *web crawling* ou tendo o nome de ficheiro igual ao URL contêm caracteres problemáticos para a análise por parte de outras ferramentas usadas ao longo do sistema implementado. Desde modo, numa primeira fase foi implementado um comando para a eliminação de todos os ficheiros que não tivessem a extensão .html. Nota, para o facto do *web crawler* atribuir automaticamente a extensão .html para páginas *web* navegadas com URL sem a extensão do ficheiro, por isso, páginas *web* construídas numa *framework* MVC, por exemplo, em que normalmente não aparece a extensão, ficam salvaguardadas.

De seguida, foi feita a substituição dos caracteres "'", "'", "º"e "\ "por "_", no nome dos ficheiros e das directorias.

4.2.3.1 Armazenamento e Redundância de Informação

As informações das páginas *web* alcançadas pelo *web crawler* são uma informação valiosa para o sistema. Um *script* escrito em Python analisa o *log* do *crawling* e extrai a data da extração, a localização do ficheiro no servidor local e no servidor remoto (URL). Os dados recolhidos são inseridos na base de dados na tabela com informação da página *web* (caso a informação ainda não exista) e na tabela com os registos das datas de recolha.

De seguida, uma análise aos registos na base de dados, verifica para cada documento descarregado no ciclo de *crawling* mais recente se o mesmo já foi processado anteriormente. Caso já tenha existido processamento da página *web* em questão, o ficheiro é eliminado, não prosseguindo para a próxima fase.

4.3 Análise das Páginas Web

Aos documentos descarregados é necessário remover *tags* e outros elementos para obtenção do corpo do texto.

Para a extração da informação da página *web* utilizou-se o Beautiful Soup[1], uma biblioteca de Python, popular pelos seus métodos simples para navegação, pesquisa e modificação de uma árvore de análise. A criação de uma aplicação não necessita de muito código.

O procedimento seguido consistiu na leitura dos novos documentos provenientes do *web crawling*, análise do HTML - com o auxílio do lxml parser -, remoção dos elementos *script* e *style*, obtenção do texto dentro dos restantes elementos e normalização dos espaços vazios entre palavras e linhas.

O resultado é guardado em ficheiros simples de texto numa diretoria. Os ficheiros serão utilizados para consulta pelo utilizador e para o processo de classificação.

4.4 Indexação dos Termos

Uma funcionalidade enriquecedora para o projeto é a pesquisa de documentos através de palavras-chave. Os documentos de texto simples necessitam de ter todos os seus termos indexados. A biblioteca de pesquisa Lucene foi a escolha. Escrita em Java, este sistema *open-source* permite uma pesquisa muito rápida sobre grandes quantidades de dados não estruturados, quando comparado com uma base de dados relacional. A desvantagem para esta última é que o Lucene indexa dados imutáveis, ou seja, que não espera que sofram alterações, pelo que uma nova indexação precisa ser indicada.

Após a análise das páginas HTML e remoção dos elementos estruturais, optou-se por fazer indexação de todos os documentos de novo.

A componente de pesquisa do Lucene permite o uso de operadores lógicos. Para a pesquisa de termos nos documentos achou-se por bem utilizador o operador "AND"que faz a correspondência para ambos os termos existirem no texto de um único documento.

4.5 Processo de Classificação

O processo de classificação divide-se em duas fases: pré-processamento dos dados e a chamada de um classificador. Numa primeira instância vários modelos de classificadores são treinados, como se pode ver na Secção 4.5.1. Depois, todos os documentos recolhidos pelo sistema são processados pelo classificador para que seja feita uma previsão da categoria a que o documento pertence, como descrito na Secção 4.5.2.

4.5.1 Treino

Na fase de treino um conjunto de dados previamente categorizados pelo utilizador são usados para a criação de *features* na etapa de pré-processamento. Na fase seguinte são treinados vários modelos de classificadores que são distintos pelos algoritmos usados no cálculos de previsão.

[1] https://www.crummy.com/software/BeautifulSoup/

4.5.1.1 Conjunto de Dados de Treino

Os documentos classificados (ou validados, caso numa fase mais adiante já tenha sido feita uma previsão e esta confirmada) pelo utilizador, com o auxílio da ferramenta apresentada mais à frente na Secção 4.6, são copiados para uma diretoria onde sofrem algumas transformações.

Nos documentos selecionados são removidas as quebras de linha e reduzidos os espaços em branco seguidos para um espaçamento. O nome do documentos é adicionado no início da linha de texto. Por fim, todos os documentos são fundidos num único ficheiro de texto simples.

Posterior os documentos são lidos por um *script* em Python e a informação é armazenada num *dataframe* - uma estrutura bidimensional de dados, como uma folha de cálculo -, da biblioteca Pandas. Com recurso à base de dados, para o nome do documento é identificado a categoria atribuída. O *dataframe* inicial é definido com duas colunas, o corpo do texto e a categoria.

4.5.1.2 Pré-Processamento

Ao *dataframe* criado na Secção 4.5.1.1 foram aplicados as seguintes funções de limpeza e preparação de dados:

- substituição de todo o texto por letras minúsculas;
- remoção de todos os sinais de pontuação;
- remoção de todos as terminações de pronomes possessivos da língua inglesa: "'s";
- aplicação do processo de *lemmatization*;
- remoção de *stop words*.

De seguida, para provar a qualidade dos modelos, dividiu-se o conjunto de dados em dois conjuntos de dados: treino e teste. A representação do conjunto de dados de treino é de só 5% face ao inicial porque não existem muitas observações (250). Esta operação resulta no preenchimento das variáveis: x_train e x_test com o corpo do texto (variáveis independentes), e y_train e y_test com as categorias (variáveis dependentes).

O TF-IDF (term frequency–inverse document frequency) foi usado para selecionar e utilizar as *features* que eram importantes na previsão. O algoritmo foi parametrizado para limitar às trezentas *features* ordenadas pelo inverso da frequência dos termos mais relevantes, as *features* a serem guardadas num vetor para serem usadas pelo classificador.

Por fim, todos os objetos com dados úteis foram guardados em ficheiros para posterior nas etapas seguintes. O módulo Pickle do Python foi usado para serialização dos objetos, conforme pode ser observado no código abaixo:

```
## x_train
with open('Pickles/x_train.pickle', 'wb') as output:
    pickle.dump(x_train, output)

## x_test
with open('Pickles/x_test.pickle', 'wb') as output:
    pickle.dump(x_test, output)
```

```
8
9  ## y_train
10 with open('Pickles/y_train.pickle', 'wb') as output:
11     pickle.dump(y_train, output)
12
13 ## y_test
14 with open('Pickles/y_test.pickle', 'wb') as output:
15     pickle.dump(y_test, output)
16
17 ## dataframe
18 with open('Pickles/df.pickle', 'wb') as output:
19     pickle.dump(df, output)
20
21 ## features_train
22 with open('Pickles/features_train.pickle', 'wb') as output:
23     pickle.dump(features_train, output)
24
25 ## labels_train
26 with open('Pickles/labels_train.pickle', 'wb') as output:
27     pickle.dump(labels_train, output)
28
29 ## features_test
30 with open('Pickles/features_test.pickle', 'wb') as output:
31     pickle.dump(features_test, output)
32
33 ## labels_test
34 with open('Pickles/labels_test.pickle', 'wb') as output:
35     pickle.dump(labels_test, output)
36
37 ## TF-IDF
38 with open('Pickles/tfidf.pickle', 'wb') as output:
39     pickle.dump(tfidf, output)
```

Listing 4.1: Excerto de código para guardar os objetos em pickles.

4.5.1.3 Modelos de Classificadores

Como o processo de treino do modelo não define os parâmetros para os modelos de classificadores que os têm, foi feito um *hyperparameter tuning*, de modo a avaliar conforme o conjunto de dados, os melhores valores. A Random Search Cross Validation (ver parte do código abaixo do parágrafo), que tem um tempo de processamento relativamente rápido, foi usada para determinação de alguns valores e de seguida, usou-se o Grid Search Cross Validation para uma análise mais exaustiva centrada sobre esses valores.

```
1  ##  Correr Random Search
2
3  gbc = GradientBoostingClassifier(random_state=8)
4
5  random_search = RandomizedSearchCV(estimator=gbc,
6                                     param_distributions=random_grid,
7                                     n_iter=50,
8                                     scoring='accuracy',
9                                     cv=3,
10                                    verbose=1,
11                                    random_state=8)
12
13 random_search.fit(features_train, labels_train)
```

```
14
15 ## Output
16
17 Fitting 3 folds for each of 50 candidates, totalling 150 fits
18 [Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent
       workers.
19 [Parallel(n_jobs=1)]: Done 150 out of 150 | elapsed: 47.6min finished
20 RandomizedSearchCV(cv=3, estimator=GradientBoostingClassifier(
       random_state=8),
21                    n_iter=50,
22                    param_distributions={'learning_rate': [0.1, 0.5],
23                                         'max_depth': [10, 40, None],
24                                         'max_features': ['auto', 'sqrt'
       ],
25                                         'min_samples_leaf': [1, 2, 4],
26                                         'min_samples_split': [10, 30,
       50],
27                                         'n_estimators': [200, 800],
28                                         'subsample': [0.5, 1.0]},
29                    random_state=8, scoring='accuracy', verbose=1)
30
31
32 ##  Ver melhores parametros devolvido pelo Random Search
33
34 print(random_search.best_params_)
35 print(random_search.best_score_)
36
37 ## Resultado
38
39 {'subsample': 1.0, 'n_estimators': 800, 'min_samples_split': 30, '
       min_samples_leaf': 4, 'max_features': 'sqrt', 'max_depth': None, '
       learning_rate': 0.1}
40 0.7637130801687765
```

Listing 4.2: Excerto de código de Random Search Cross validation no modelo de classificação Gradient Boosting.

De seguida, os modelos de classificadores são treinados, usando o método *fit*, e utilizando o conjunto de dados de treino. Neste ponto, o classificador está preparado para fazer previsões. A serialização do classificador é feita para ficheiro.

Por fim, os classificadores são usados para treinar o conjunto de dados de treino e apuram-se as precisões obtidas. O classificador com melhores resultados será utilizado nas previsões feitas pelo sistema.

4.5.2 Previsão

Nesta fase, são utilizados o modelo de classificação treinado com melhores resultados e o modelo de TF-IDF.

Uma leitura à base de dados devolve todos os caminhos dos ficheiros de texto existentes. O *script* de Python, para cada ficheiro, guarda numa variável o conteúdo do documento e faz uma limpeza e preparação dos dados idêntica à enunciada na Secção 4.5.1.2. O resultado é avaliado pelo classificador que devolve a previsão da categoria para o texto. Os resultados são usados para atualizar na base de dados a categoria prevista pelo sistema desenvolvido.

4.6 Ferramenta de Visualização

A interface de comunicação entre o sistema e o utilizador encontra-se desenvolvido sobre a forma de um *website*. A *framework web* Bootstrap 4 foi utilizada para um rápido e fácil desenvolvimento das páginas *web* com uma moderna aparência da formatação de texto, tabelas e de outros elementos da interface.

De encontra aos requisitos foram desenvolvidas as seguintes páginas:

4.6.1 Login AUTOCLIPPING (index.php)

Nesta página o utilizador tem de fazer a sua autentificação. Um formulário onde devem ser inseridos o utilizador e a respetiva palavra-passe é apresentado. Após a submissão dos dados um pedido AJAX a uma outra página *web*, com recurso à linguagem de programação PHP, faz a validação na base de dados e guarda o utilizador numa variável de sessão.

Após a validação, caso as credenciais estejam corretas o utilizador é direcionado para a página inicial, senão é pedido que as credenciais sejam novamente inseridas.

Para qualquer página *web* em que exista a tentativa de acesso é verificado se a variável de sessão com o utilizador está definida. Caso esta variável não exista, o utilizador é reencaminhado para esta página de autentificação.

4.6.2 Homepage AUTOCLIPPING (home.php)

A página inicial deve apresentar sobre a forma de hiperligações as páginas de interesse para o utilizador:

- Documents and Classifications;
- News by Crawling;
- New News by Crawling;
- Personal Repository;
- Statistics;
- Crawling Seeds.

A opção de terminar a sessão deve estar também presente. Através da eliminação da variável de sessão, o utilizador deixa de estar autentificado.

4.6.3 Documents and Classifications (documents_classifications.php)

Todos os documentos recolhidos pelo processo de *web crawling* devem ser apresentados nesta página sobre a forma de uma tabela e com informação do tópico, *website*, URL, data da última recolha e hiperligação para o documento de texto analisado.

A categoria prevista pelo classificador treinado deve aparecer também nesta página, bem como uma *checkbox* em que o utilizador deve confirmar se está de acordo com a atribuição desse valor. Caso a categoria prevista esteja incorreta, uma outra coluna com um menu

dropdown, permite ao utilizador selecionar de entre todas as categorias predefinidas para aquele tópico a que este considera correta. Os valores da*checkbox* e do *dropdown* são armazenados na base de dados e lidos pelo código HTML caso existam sendo definidos por defeito, pelo que a qualquer momento todos os utilizadores podem ver estas modificações. A categoria prevista pode sofrer alterações porque o classificador é recriado com frequência como parte do processo de aprendizagem ativa, por isso, caso o categoria prevista seja validada pelo utilizador, esse valor é copiado para a coluna das categorias definidas pelo utilizador. Uma coluna na base de dados é responsável por guardar a indicação de porque a coluna das categorias definidas pelo utilizador foi modificada, para efeitos de estatística.

Nesta página uma coluna que permite adicionar o registo selecionado ao repositório pessoal do utilizador. Documentos já adicionados têm a opção desativada.

Para a maioria das colunas, com recurso a funções da biblioteca JQuery, existe um campo de entrada de texto abaixo do cabeçalho da coluna que o utilizador pode utilizar para filtrar os registos. Ao clicar no cabeçalho da coluna são ainda ordenados os valores da coluna alternadamente por ordem crescente ou decrescente.

Por fim, existe um campo de entrada de texto onde o utilizador pode especificar termos que devem estar nos documentos apresentados. Os termos são passados por um pedido AJAX para uma página web em PHP que faz uma chamada ao software Lucene que devolve os documentos identificados. Os identificadores dos documentos são devolvidos por JSON, e a página com recurso de métodos da biblioteca JQuery passa a apresentar só os documentos que contêm todos os termos introduzidos.

4.6.4 News by Crawling (news_crawling.php)

Uma tabela é apresentada nesta página com a informação do tópico, *website*, data de recolha, quantidade de notícias recolhidas - informação obtida através da contagem de registos por data e *website* - e uma hiperligação que redireciona para a página referida na Secção 4.6.3 e apresenta o conjunto de notícias selecionado.

4.6.5 New news by Crawling (new_news_crawling.php)

Página semelhante à página apresentada na Secção 4.6.4 com a diferença que em vez da quantidade de notícias recolhidas é antes apresentado a quantidade de novas notícias recolhidas, ou seja, são contabilizadas notícias que ainda não tinham sido anteriormente recolhidas e processadas.

4.6.6 Personal Repository (personal_repository.php)

Cada utilizador tem um repositório onde estão as notícias guardadas pelo mesmo. Esta página é semelhante à página existente na Secção 4.6.3 com as diferenças que só são apresentas as notícias salvas pelo utilizador e a opção de adição do documento ao repositório é substituída pela opção de remoção do documento do repositório. Com auxílio do JQuery, o utilizador quando aciona o botão de remover o documento, para além deste ser eliminado na base de dados, é eliminado da tabela inicialmente apresentada sem necessidade de refrescar a página.

4.6.7 Statistics (stats.php)

Nesta página existe um menu *dropdown* com os *websites* recolhidos que depois de selecionado um é apresenta uma tabela com dados estatísticos para esse *website*, com base nas informações armazenadas na base de dados:

- número de categorias confirmadas;
- número de categorias não confirmadas;
- número de categorias bem previstas;
- número de categorias mal previstas;
- data do último ciclo de web crawling;
- número de notícias recolhidas no último ciclo de web crawling.

4.6.8 Crawling Seeds (crawling_seeds.php)

Por último, existe uma página onde é feita a gestão dos *seed* URL usados pelo *web crawling*. Numa tabela são apresentados o *website*, *seed URL* e um botão que permite remover o registo a lista de crawling. Ao utilizador ainda é dado a opção de inserir num campo de texto um novo *seed* URL. Após submissão é aplicada uma função para obtenção do *website* e tanto o *seed* URL como o *website* são armazenado na base de dados para posterior análise por parte do processo de *web crawling*.

Capítulo 5

Resultados

Neste capítulo vão ser apresentados as métricas usadas no acompanhamento do pré-processamento do conjunto de dados de treino e nos testes feitos aos modelos de classificação treinados na escolha do classificador com melhores resultados. Por fim, a estrutura *web* desenvolvida para ser a ponte entre o utilizador registado e o sistema é apresentada.

5.1 Recolha de Dados

Apesar do sistema permitir a utilização de diversos tópicos, os documentos analisados são exclusivamente relacionados com o assunto Erasmus+. O processo de *web crawling* devolveu um total de 3772 documentos, num período de 3 meses, para os *seed* URL mencionados na Secção 3.3. No processo de treino do modelo de classificação foi utilizado um conjunto de dados com 250 documentos classificados, havendo um cuidado na distribuição equitativa entre categorias como pode ser visto na Figura 5.1,

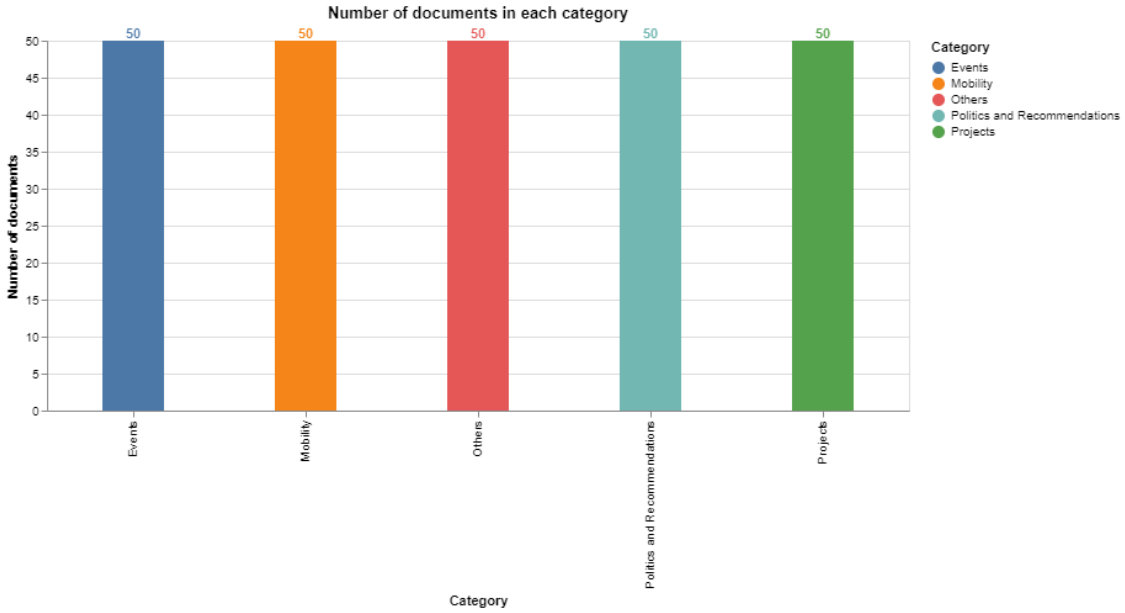

Figura 5.1: Diagrama do número de documentos de treino para cada categoria.

Analisando o diagrama de extremos e quartis da Figura 5.2 observa-se que a categoria Outros tem documentos de dimensão mais reduzida, comparativamente com as outras categorias, provavelmente por a maioria do conteúdo serem páginas sem valor para o projeto desenvolvido, como páginas de autentificação, menus, páginas de contactos, entre outros. A categoria Projetos é a que tem destacadamente os documentos mais volumosos, muito devido ao facto de incluir muitas informações sobre os projetos, como propósito, a quem se destina, requisitos, entre outros. Na criação das *features* os valores foram normalizados para para se evitar que palavras que aparecem mais vezes porque o documento é maior influenciem o processo de classificação.

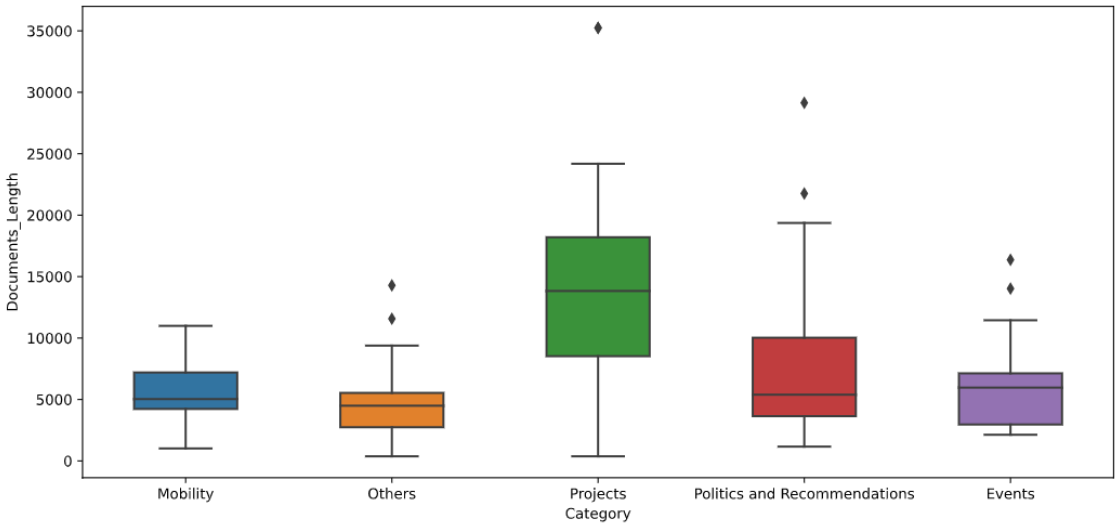

Figura 5.2: Diagrama de extremos e quartis com a dimensão dos documentos de treino para cada categoria.

5.2 Pré-Processamento

As técnicas aplicadas no pré-processamento para redução do conjunto de dados de treino resultaram numa diminuição de 13% no número de caracteres e de 26% no número de palavras. Conforme pode ser observado na Tabela 5.1 a maior redução aconteceu na remoção das *stop words*.

Tabela 5.1: Impacto na redução da dimensão do conjunto de dados de treino.

Variação ao longo das técnicas de pré-processamento aplicadas.

	Documento Original	Sinais de Pontuação	Terminações de Pronomes Possessivos	Lemmatization	Stop Words	ΔTotal
Nº de Palavras	271661	270666	270666	270666	199812	-71849 (-26%)
Nº de Caracteres	1856126	1837270	1836434	1790514	1590264	-265862 (-13%)

Depois da redução da dimensão os documentos foram representados num vetor TF-IDF. O modelo representa diminui o peso dos valores que aparecem com maior frequência ao mesmo tempo que aumenta o peso dos valores que aparecem mais raramente. Através da biblioteca de Python chi2 que faz uso da do modelo estatístico de distribuição x^2, muito usado em estatística inferencial, encontraram-se os termos que maior correlação para cada categoria. O resultado pode ser observado na Tabela 5.2. Os termos com o tamanho de uma palavra são chamados de unigramas e os termos que aparecem mais vezes de forma seguida e com o tamanho de duas palavras são chamados de bigramas.

Tabela 5.2: N-gramas mais correlacionados com cada categoria.

	Eventos	Mobilidade	Outros	Políticas e Recomendações	Projetos
Unigramas	application	close	student	adult	eac
	eurydice	use	section	secondary	11
	faqs	teach	international	early	emjmd
	apply	abroad	esn	childhood	a03
	eit	categories	categories	teachers	cest
Bigramas	erasmus student	higher education	student network	secondary education	erasmus key
	model eit	learn mobility	erasmus student	early childhood	cest close

5.3 Classificadores

O conjunto de algoritmos de classificação treinados foi avaliado com recurso a algumas métricas: precisão, exatidão, *recall*, *f1-score*, RMSE e MAE. O conjunto de dados de teste foi analisado pelos modelos de classificação treinados concebidos sobre o conjunto de dados de treino. Os valores encontra-se descritos na Tabela 5.3 e são a média ponderada para cada categoria. Com os valores mais altos para precisão, exatidão, *recall* e *f1-score*, ao mesmo tempo o modelo previsão com menores taxas de erro (utilizando o RMSE e MAE), o classificador que se destaca pelos melhores resultados baseia-se no algoritmo de Gradient Boosting. A matriz de confusão deste algoritmo (ver Figura 5.3) mostra que a maioria das categorias foi corretamente classificada pois a maioria dos pontos encontra-se representados na diagonal.

Tabela 5.3: Resultado dos testes feitos com os classificadores.

	Precisão	Exatidão	Recall	F1-score	RMSE	MAE
K- Nearest Neighbour (KNN)	0.69	0.73	0.69	0.70	1.18	0.62
Multinomial Naive Bayes	0.54	0.56	0.54	0.54	1.71	1.07
Gradient Boosting	0.85	0.88	0.85	0.85	0.39	0.15
Random Forest	0.77	0.83	0.77	0.77	1.18	0.46
Support Vector Machines (SVM)	0.62	0.73	0.62	0.65	1.88	1.08
Logistic Regression	0.69	0.83	0.69	0.72	1.086	1.00
Long Short Term Memory (LSTM)	0.60					

5.4 Ferramenta de Visualização

A interface de comunicação entre o o utilizador e o sistema foi desenvolvido um *website*.

Na Figura 5.4 mostra a página inicial com os atalhos para os recursos disponíveis para o utilizador.

Na Figura 5.5 um recorte da página que apresenta todos os documentos recolhidos, bem como a previsão feita pelo classificador desenvolvido e a sua validação pelo utilizador. Destaque para a quarta linha da figura em que se pode ver um registo que foi classificado pelo utilizador como fazendo referência à categoria Projetos, mas que o classificador desenvolvido classificou erradamente como Evento.

Os documentos recolhidos são agrupados em cada ciclo de *crawling* e é extraída informação quanto ao número de documentos conforme mostra na Figura 5.6 e quanto ao número de novas notícias conforme se vê na Figura 5.7.

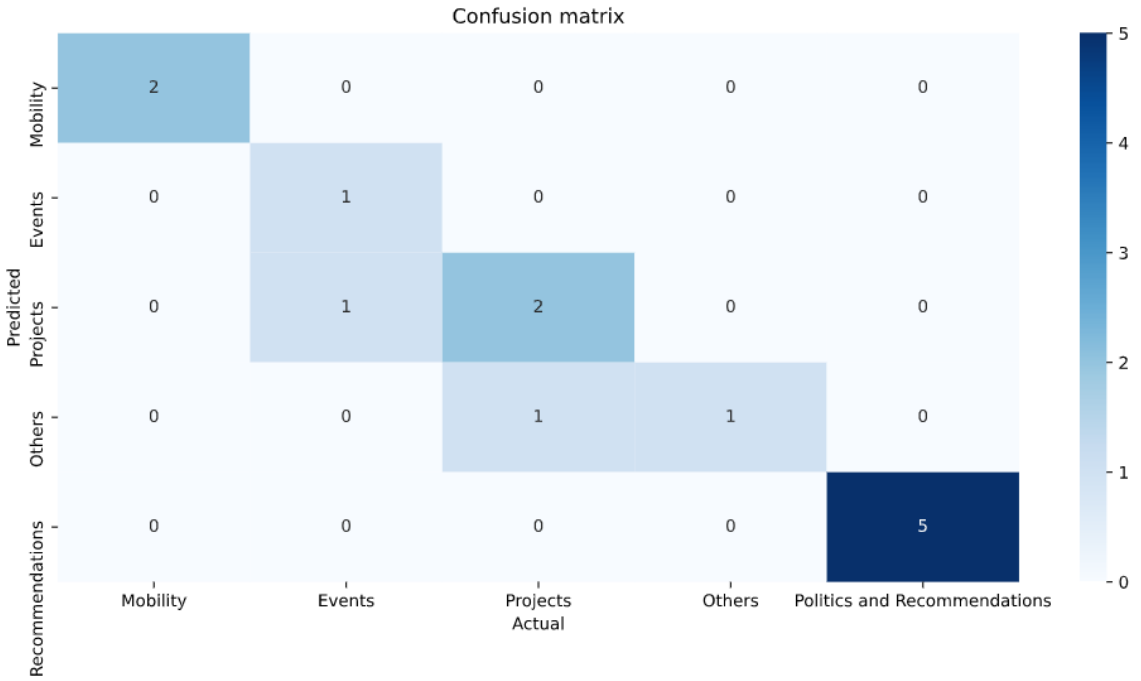

Figura 5.3: Matriz de confusão dos testes ao modelo de classificação de Gradient Boost.

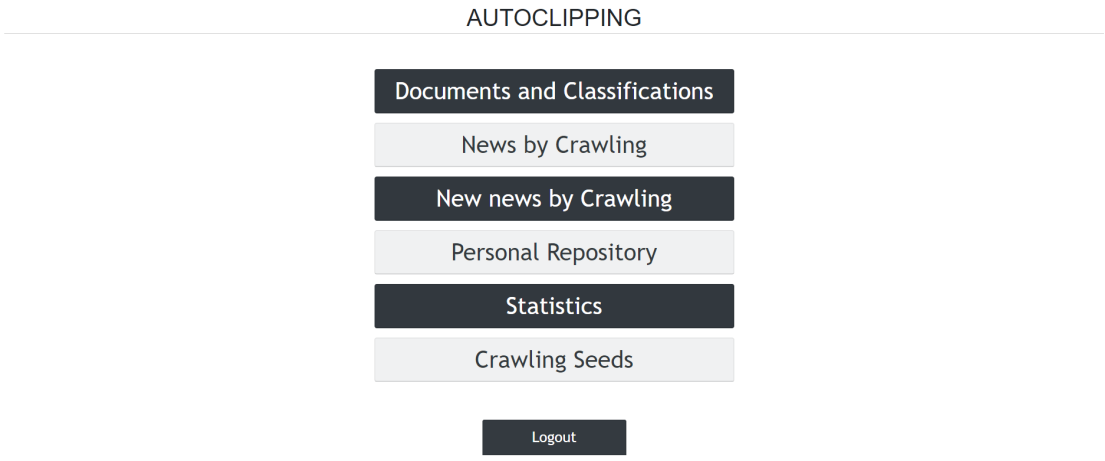

Figura 5.4: Página Web Homepage AUTOCLIPPING.

Os dados sobre a recolha de documentos e a sua classificação são trabalhados para a produção de estatística. Na Figura 5.8 encontra-se o exemplo desta funcionalidade aplicada a um determinado *website*.

O utilizador tem a opção de guardar as notícias apresentadas num repositório que é exclusivo de cada membro. O aspeto da página *web* é apresentado na Figura 5.9.

Por fim, os *seed* URL utilizados pelo processo de *web crawling* podem ser consultados, eliminados ou adicionados na página web apresentada na Figura 5.10.

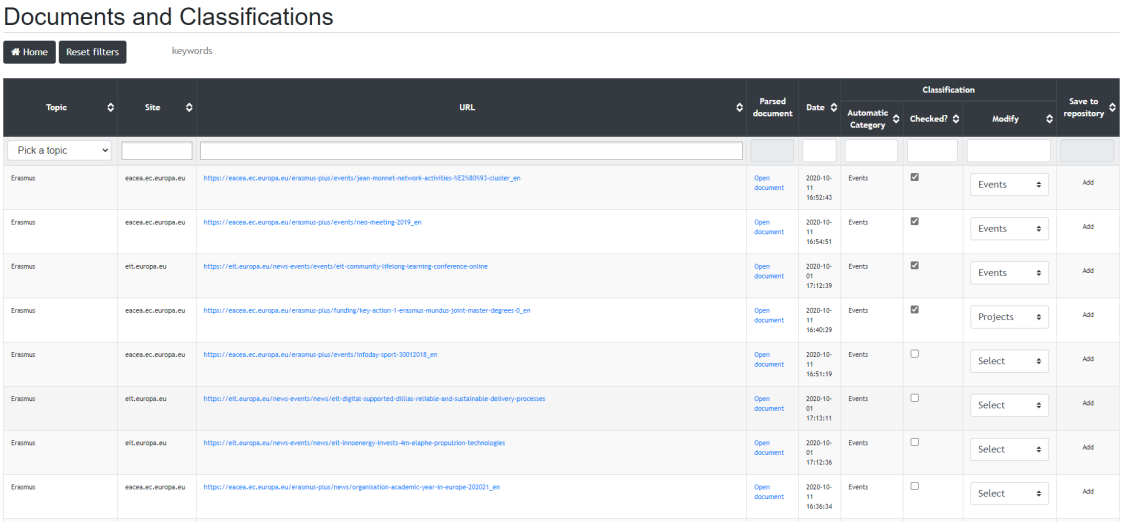

Figura 5.5: Página Web Documents and Classifications.

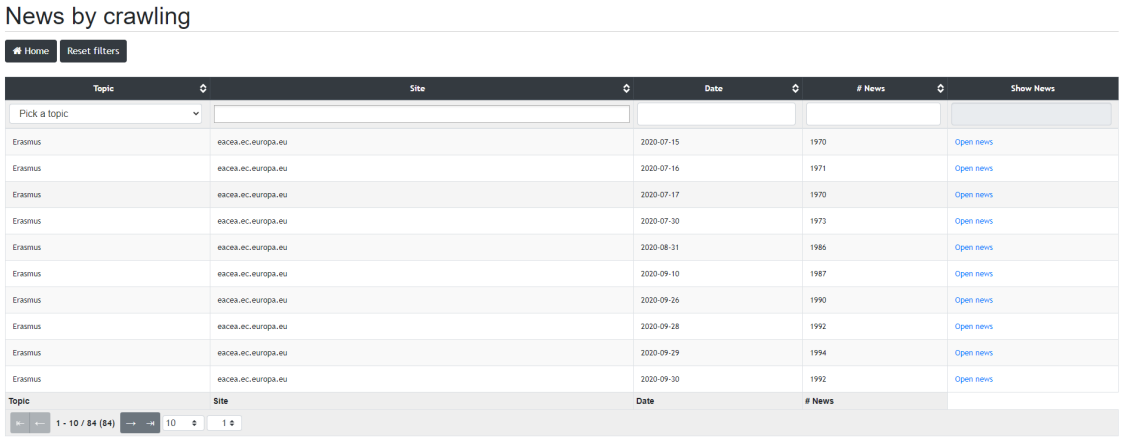

Figura 5.6: Página Web News by Crawling.

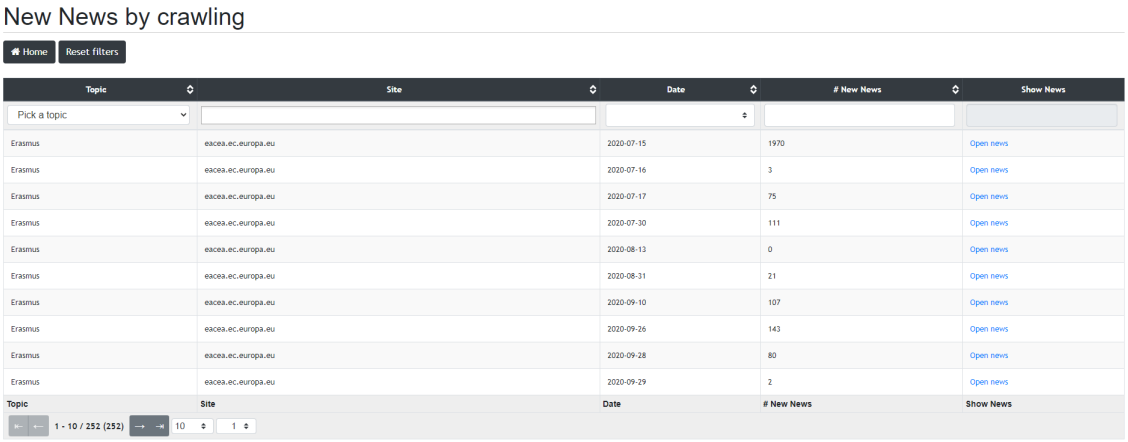

Figura 5.7: Página Web New News by Crawling.

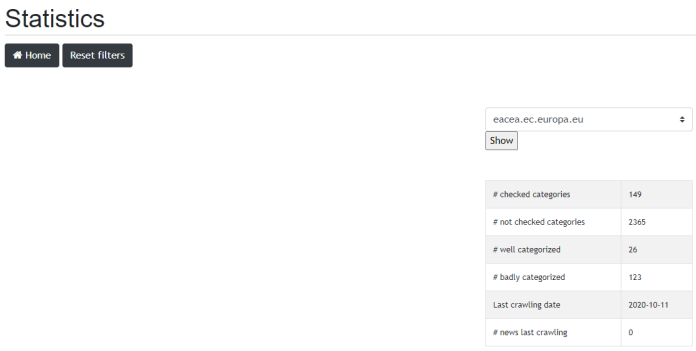

Figura 5.8: Página Web Statistics.

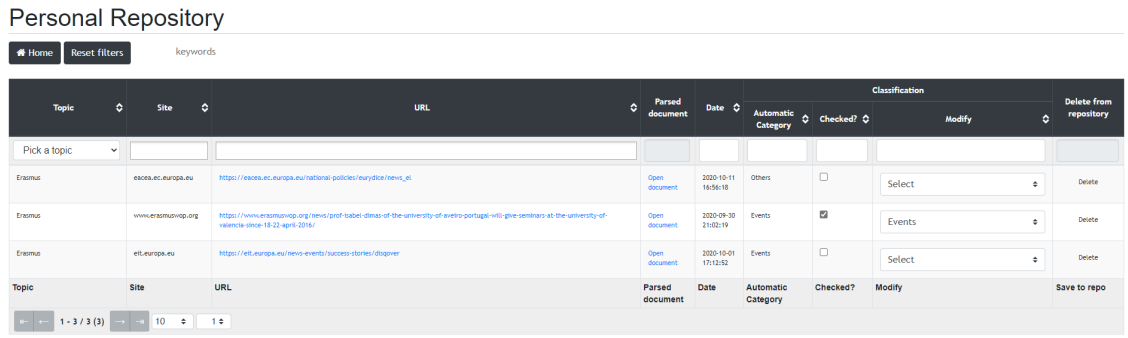

Figura 5.9: Página Web Personal Repository.

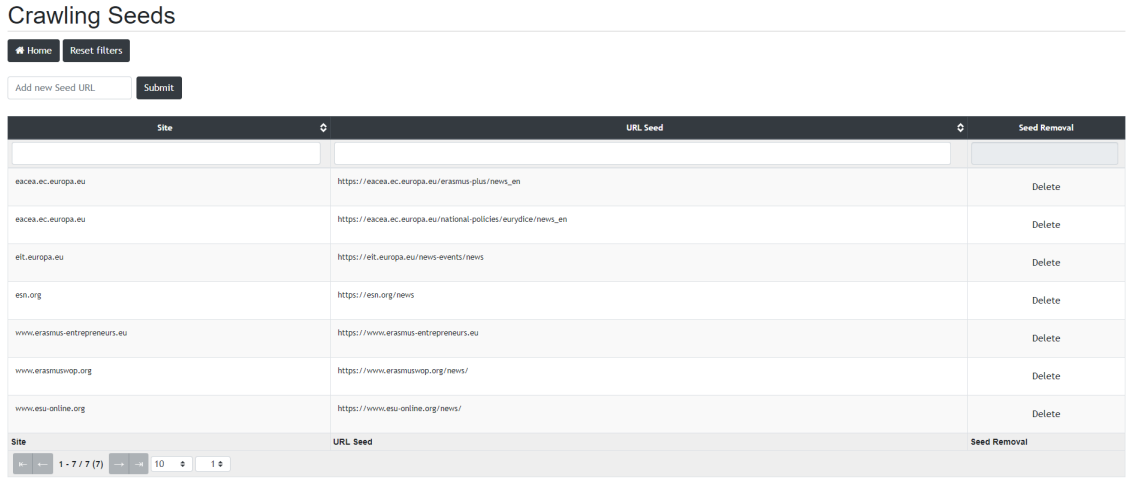

Figura 5.10: Página Web Crawling Seeds.

Capítulo 6

Conclusão

Cada vez mais as instituições apostam no uso de soluções tecnológicas para a substituição de tarefas até agora feitas pelo ser humano. Neste sentido, a European Association of ERASMUS Coordinators mostrou interesse numa solução capaz de recolher notícias e as agrupar em categorias. Tarefa atribuída até agora a membros desta associação.

O projeto proposto é ambicioso: ter um *software* informático capaz de interpretar informações escritas em linguagem humana e dar uma resposta nessa mesma linguagem.

As notícias recolhidas resultaram de um sistema que teve de ser preparado para navegar a partir de uma página indicada e daí para outras páginas *web* mencionadas no contexto, e assim sucessivamente, fazendo uma cópia *offline* a ser guardada localmente. Para os ficheiros recolhidos foi necessário distinguir os dados que compunham o corpo da notícia.

Ao corpo de notícias foram aplicadas técnicas para a extração de características (*features*) próprias de cada categoria. Através desse conhecimento uma seleção de modelos de classificação foi usada para identificar para novos documentos o tema do seu conteúdo. Um conjunto de métricas de avaliação foi usado sobre os resultados, permitindo ter diferentes formas de comparar os resultados previstos e os resultados verdadeiros. Os valores obtidos da avaliação possibilitaram determinar o classificador com melhor desempenho: modelo de *Grandient Boosting*.

Por fim, foi criada uma plataforma para o utilizador final interagir com o sistema. Um *website* foi a decisão tomada para apresentar aos membros da associação os documentos recolhidos e as respetivas classificações.

6.1 Objetivos alcançados

O sistema foi implementado com todos os objetivos cumpridos. A proposta inicial propunha um sistema capaz de recolher notícias de forma automática agrupando-as depois em diferentes diretorias conforme a sua categoria.

O sistema implementado recolhe de forma automática todas as páginas *web* a partir de *seeds* URL que o utilizador define, com exceção de páginas de estilo e de *scrips* porque não têm informação relevante. Os *seed* URL podem ser consultados, adicionados e eliminados numa das páginas *web* da plataforma *web* desenvolvida.

Ás notícias recolhidas estão originalmente no formato HTML com *tags* e outros elementos usados na construção destas infraestruturas *web*. Um *script* converte o ficheiro HTML para um ficheiro de texto simples e limpa esses elementos. Tanto o URL associado à notícia

como o ficheiro simples de texto são disponibilizados no *website*. Não é propriamente um sistema de diretorias como era proposto, mas facilmente utilizado um dos filtros existentes na página *web* é possível selecionar a categoria e são apresentados só documentos desse tipo.

A categorização dos documentos foi bem conseguida, através de um modelo de classificação de aprendizagem automática treinado para este tipo de dados. O classificador foi criado com baixas taxas de erro.

Uma das grandes vantagens de ter este sistema a tratar deste tipo de tarefas é o tempo poupado ao utilizador na procura, análise e extração de notícias. A instituição poderá alocar o membro neste tempo poupado noutros projetos.

6.2 Trabalho Futuro

Os algoritmos de classificação de texto treinados não fazem sempre a melhor previsão. As causas de classificações erradas podem ser: o conjunto de dados de treino deveria ser maior para um melhor apuramento das *features* de cada categoria; a seleção das *features* deveria ser testado com outros tipos de técnicas como, por exemplo, Count Words.

Outra coisa, que ficou pouco desenvolvida foi a possibilidade de serem usados outros tópicos no sistema desenvolvido. Usou-se ao longo de toda a solução o tópico "Erasmus+". Todas as notícias apesar de terem diferentes categorias, pertencem a este tópico. No entanto, a solução inicialmente foi desenvolvida para suportar vários tópicos. Tanto na base de dados como na visualização dos documentos é possível ver disponibilizado o campo para o tópico, no entanto não foi associado aos *seed* URL, nem às páginas *web* transferidas recursivamente.

Fica a faltar a opção de adicionar ou remover categorias associadas a um tópico no *website*. Até agora essa inserção foi feita manualmente.

Bibliografia

Adnan, Kiran e Rehan Akbar (2019). «An analytical study of information extraction from unstructured and multidimensional big data». Em: *Journal of Big Data* 6.1. issn: 21961115. doi: 10.1186/s40537-019-0254-8. url: https://link.springer.com/article/10.1186/s40537-019-0254-8.

Apoio à reforma das políticas | Erasmus+ (2020). url: https://ec.europa.eu/programmes/erasmus-plus/opportunities/support-policy-reform%7B%5C_%7Dpt (acedido em 20/09/2020).

Aryal, Sunil et al. (2019). «A new simple and effective measure for bag-of-word inter-document similarity measurement». Em: *CoRR* abs/1902.03402. arXiv: 1902.03402. url: http://arxiv.org/abs/1902.03402.

Banu, G. Rasitha e VK Chitra (2015). «A Survey of Text Mining Techniques and\nApplications». Em: *Internacional Jornal and Technology (IJIET)* 2.2319-1058.

Bolandraftar, Mohammad, Sadegh Bafandeh e Imandoust And. *Application of K-nearest neighbor (KNN) approach for predicting economic events theoretical background Application of K-Nearest Neighbor (KNN) Approach for Predicting Economic Events: Theoretical Background*. Rel. téc., pp. 605–610. url: www.ijera.com.

Chai, T e R R Draxler (2014). «Root mean square error (RMSE) or mean absolute error (MAE)? -Arguments against avoiding RMSE in the literature». Em: *Geoscientific Model Development* 7.3, pp. 1247–1250. issn: 19919603. doi: 10.5194/gmd-7-1247-2014. url: www.geosci-model-dev.net/7/1247/2014/.

Chen, Jingnian et al. (2009). «Feature selection for text classification with Naïve Bayes». Em: *Expert Systems with Applications* 36.3 PART 1, pp. 5432–5435. issn: 09574174. doi: 10.1016/j.eswa.2008.06.054.

Cooley, R., B. Mobasher e J. Srivastava (1997). «Web mining: Information and pattern discovery on the World Wide Web». Em: *Proceedings of the International Conference on Tools with Artificial Intelligence*, pp. 558–567. issn: 10636730. doi: 10.1109/tai.1997.632303.

EAEC Network - About EAEC (2020). url: http://www.eaecnet.com/index.php?id=7 (acedido em 20/09/2020).

Gooch, Daniel (2011a). «Communications of the ACM». Em: *XRDS: Crossroads, The ACM Magazine for Students* 18.2, p. 6. issn: 15284972. doi: 10.1145/2043236.2043240.

– (2011b). «Communications of the ACM». Em: *XRDS: Crossroads, The ACM Magazine for Students* 18.2, p. 6. issn: 15284972. doi: 10.1145/2043236.2043240.

Goutte, Cyril e Eric Gaussier (2005). «A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation». Em: *Lecture Notes in Computer Science* 3408, pp. 345–359. issn: 03029743. doi: 10.1007/978-3-540-31865-1_25.

Gurusamy, Vairaprakash e Associate Professor. *Preprocessing Techniques for Text Mining*. Rel. téc.

High, Rob. *Front cover The Era of Cognitive Systems: An Inside Look at IBM Watson and How it Works*. Rel. téc.

Ikonomakis, Emmanouil K et al. *Text Classification Using Machine Learning Techniques Educational Data Mining View project Active Learning View project Text Classification Using Machine Learning Techniques*. Rel. téc. url: https://www.researchgate.net/publication/228084521.

Iyyer, Mohit et al. (2015). «Deep unordered composition rivals syntactic methods for text classification». Em: *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*. Vol. 1, pp. 1681–1691. isbn: 9781941643723. doi: 10.3115/v1/p15-1162.

Jing, Li Ping, Hou Kuan Huang e Hong Bo Shi (2002). «Improved feature selection approach TFIDF in text mining». Em: *Proceedings of 2002 International Conference on Machine Learning and Cybernetics*. Vol. 2, pp. 944–946. isbn: 0780375084. doi: 10.1109/icmlc.2002.1174522.

Joachims, Thorsten. *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*. Rel. téc.

Jourdan, Zack, R. Kelly Rainer e Thomas E. Marshall (2008). «Business intelligence: An analysis of the literature». Em: *Information Systems Management* 25.2, pp. 121–131. issn: 10580530. doi: 10.1080/10580530801941512.

Justo, Andreia (2018). *O que é um projeto? Quais os principais tipos de projeto?* url: https://www.euax.com.br/2018/08/o-que-e-um-projeto/ (acedido em 01/10/2020).

Know, Did You (2012). «A Brief History of Text Messaging». Em: *BeyeNetwork, October* 30, pp. 5–6. url: http://www.b-eye-network.com/view/6311.

Martin, Vanessa (2003). *Manual Prático de Eventos*. Editora Atlas, p. 277. isbn: 9788522433940.

Mulins, Matt e Matt Mullins (2008). *Information extraction in text mining Information Extraction in Text Mining Information Extraction in Text Mining*. Rel. téc. url: https://cedar.wwu.edu/computerscience%7B%5C_%7Dstupubshttps://cedar.wwu.edu/computerscience%7B%5C_%7Dstupubs/4.

Nahm, Un Yong e Raymond J Mooney (2002). *Text Mining with Information Extraction*. Rel. téc. url: www.aaai.org.

O Tutorial Python — documentação Python 3.8.6 (2020). url: https://docs.python.org/pt-br/3/tutorial/ (acedido em 04/10/2020).

O.A. (2016). *PostgreSQL: About*. url: https://www.postgresql.org/about/ (acedido em 04/10/2020).

Plisson, Joël, Nada Lavrac e Dr. Dunja Mladenić (2004). «A rule based approach to word lemmatization». Em: *Proceedings of the 7th International Multiconference Information Society (IS'04)*, pp. 83–86. url: http://eprints.pascal-network.org/archive/00000715/.

Popp, R. K. (2014). «Information, Industrialization, and the Business of Press Clippings, 1880-1925». Em: *Journal of American History* 101.2, pp. 427–453. issn: 0021-8723. doi: 10.1093/jahist/jau373. url: https://academic.oup.com/jah/article-lookup/doi/10.1093/jahist/jau373.

Projetos de mobilidade nos domínios da educação, formação e juventude | Erasmus+ (2020). url: https://ec.europa.eu/programmes/erasmus-plus/programme-guide/part-b/three-key-actions/key-action-1/mobility-education-training-youth%7B%5C_%7Dpt (acedido em 20/09/2020).

S. Vijayarani, J. Ilamathi e Nithya (2015). «Preprocessing Techniques for Text Mining-An Overview Privacy Preserving Data Mining View project». Em: *International Journal of Computer Science & Communication Networks* 5.1, pp. 7–16. url: https://www.researchgate.net/publication/339529230.

Santos, Russ delos (2019). *Text Classification. by Russ delos Santos | by RAX Automation Suite | Medium*. url: https://medium.com/@raxsuite/text-classification-69f60e0e2ce5 (acedido em 14/10/2020).

Talib, Ramzan et al. (2016a). «Text Mining: Techniques, Applications and Issues». Em: *International Journal of Advanced Computer Science and Applications* 7.11, pp. 414–418. issn: 2158107X. doi: 10.14569/ijacsa.2016.071153.

– (2016b). «Text Mining: Techniques, Applications and Issues». Em: *International Journal of Advanced Computer Science and Applications* 7.11, pp. 414–418. issn: 2158107X. doi: 10.14569/ijacsa.2016.071153.

Text Mining Software, SAS Text Miner | SAS (2020). url: https://www.sas.com/en%7B%5C_%7Dus/software/text-miner.html (acedido em 20/02/2020).

Text Mining: What is text mining and how it can be useful in Analytics (2020). url: https://www.edupristine.com/blog/text-mining-overview (acedido em 14/01/2020).

TextAnalyst - new text mining solution from Megaputer - Megaputer Intelligence (2020). url: https://www.megaputer.com/press/textanalyst-new-text-mining-solution-from-megaputer/ (acedido em 20/02/2020).

Watson Natural Language Understanding - Features | IBM (2020). url: https://www.ibm.com/cloud/watson-natural-language-understanding/details (acedido em 20/02/2020).

Wen, Qinghua et al. (2010). «Automatic stock decision support system based on box theory and SVM algorithm». Em: *Expert Systems with Applications* 37.2, pp. 1015–1022. issn: 09574174. doi: 10.1016/j.eswa.2009.05.093. url: http://dx.doi.org/10.1016/j.eswa.2009.05.093.